

AD-A124 884

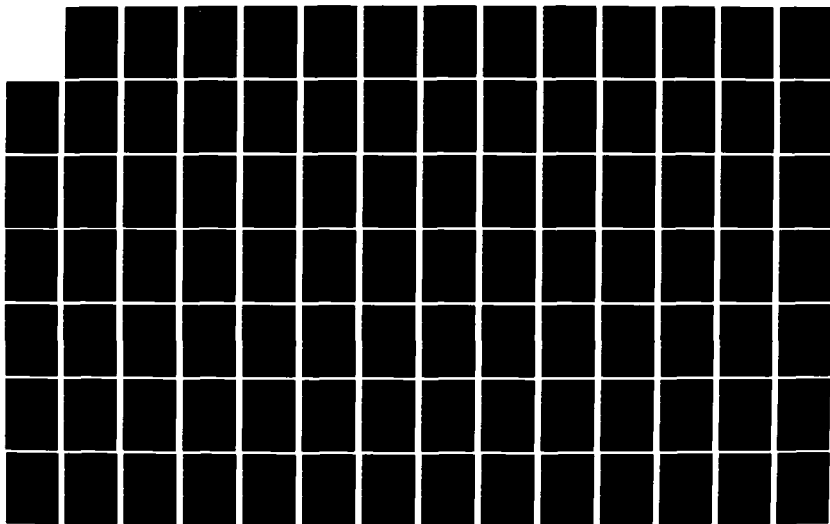
ENHANCED TRACKING OF AIRBORNE TARGETS USING A
CORRELATOR/KALMAN FILTER(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING
P P MILLNER DEC 82 AFIT/GE/EE/82D-50

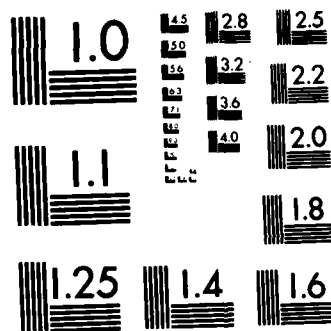
1/4

UNCLASSIFIED

. F/G 12/1

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

AD A124884



ENHANCED TRACKING OF AIRBORNE TARGETS
USING A CORRELATOR/KALMAN FILTER

THESIS

AFIT/GE/EE/82D- 50

Paul P. Millner
CPT US Army

This document has been approved
for public release and sale; its
distribution is unlimited.

DEPARTMENT OF THE AIR FORCE
AIR UNIVERSITY (ATC)

AIR FORCE INSTITUTE OF TECHNOLOGY

Wright-Patterson Air Force Base, Ohio

DTIC
ELECTE
FEB 24 1983

A

DTIC FILE COPY

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE (When Data Entered)

REPORT DOCUMENTATION PAGE		READ INSTRUCTIONS BEFORE COMPLETING FORM
1. REPORT NUMBER AFIT/GE/EE/82D-50	2. GOVT ACCESSION NO. AD-4124884	3. RECIPIENT'S CATALOG NUMBER
4. TITLE (and Subtitle) ENHANCED TRACKING OF AIRBORNE TARGETS USING A CORRELATOR/KALMAN FILTER		5. TYPE OF REPORT & PERIOD COVERED MS Thesis
7. AUTHOR(s) Paul P. Millner, Captain, US Army		6. PERFORMING ORG. REPORT NUMBER
9. PERFORMING ORGANIZATION NAME AND ADDRESS Air Force Institute of Technology (AFIT-EN) Wright-Patterson AFB, Ohio 45433		8. CONTRACT OR GRANT NUMBER(s)
11. CONTROLLING OFFICE NAME AND ADDRESS		10. PROGRAM ELEMENT, PROJECT, TASK AREA & WORK UNIT NUMBERS
14. MONITORING AGENCY NAME & ADDRESS (if different from Controlling Office)		12. REPORT DATE December 1982
		13. NUMBER OF PAGES 357
		15. SECURITY CLASS. (of this report) Unclassified
		15a. DECLASSIFICATION/DOWNGRADING SCHEDULE
16. DISTRIBUTION STATEMENT (of this Report) Approved for public release; distribution unlimited		
17. DISTRIBUTION STATEMENT (of the abstract entered in Block 20, if different from Report)		
18. SUPPLEMENTARY NOTES Approved for public release; LAW AFR 190-17 4 JAN 1982 LARRY E. WOLAVER Dean for Research and Professional Development Air Force Institute of Technology (ATC) Wright-Patterson AFB OH 45433		
19. KEY WORDS (Continue on reverse side if necessary and identify by block number) Kalman Filter Adaptive Estimation Target Tracking Forward Looking Infrared Systems Correlators Fast Fourier Transforms		
20. ABSTRACT (Continue on reverse side if necessary and identify by block number) Over the past four years considerable work has been accomplished at the Air Force Institute of Technology to improve the tracking capability of the high energy laser weapon against airborne targets. In this research, many of the prior concepts are incorporated into a correlator/Kalman filter to develop a tracker capable of providing precise target position estimates in a dynamic short-range environment using a Foward		

UNCLASSIFIED

SECURITY CLASSIFICATION OF THIS PAGE(When Data Entered)

Looking Infrared sensor (FLIR) to provide measurement data. Digital signal processing is employed on the FLIR data to identify the underlying target intensity shape function when the target under consideration has either single or multiple "hot spots". The estimated target shape function is then used as the template in a correlation algorithm, where spatial and frequency domain correlation techniques were explored, to determine the offsets between the template and the incoming measurement. These offsets are used as "pseudomeasurements" in a linear Kalman filter which exploits knowledge of the process dynamics and statistical knowledge of the correlator errors to enhance the position estimates.

AFIT/GE/EE/82D-50

ENHANCED TRACKING OF AIRBORNE TARGETS
USING A CORRELATOR/KALMAN FILTER

THESIS

AFIT/GE/EE/82D-50

Paul P. Millner
CPT US Army

DTIC

700 2 4 1983

A

Approved for public release; distribution unlimited

ENHANCED TRACKING OF AIRBORNE TARGETS
USING A CORRELATOR/KALMAN FILTER

THESIS

Presented to the Faculty of the School of Engineering
of the Air Force Institute of Technology
Air University
in Partial Fulfillment of the
Requirements for the Degree of
Master of Science

by
Paul P. Millner, B.S.
CPT US Army
Graduate Electrical Engineering
December 1982

Accession For	
NTIS GR&I	<input checked="checked" type="checkbox"/>
DDIC TAB	<input type="checkbox"/>
Unannounced	<input type="checkbox"/>
Justification	
Distribution/	
Availability Codes	
Avail and/or	
Dist	Special

A

Approved for public release; distribution unlimited.



Preface

This study is part of a continuing effort to design a tracking system for use in a ground based laser system under development by the Air Force Weapons Laboratory. A correlator/Kalman filter which uses infrared sensor data was synthesized and tested.

I wish to express my appreciation to my thesis advisor, Dr. Peter S. Maybeck, for his expert guidance and enthusiastic support during this project.

Contents

	Page
Preface	ii
List of Figures	v
List of Tables	vii
List of Symbols	viii
Abstract	x
I. Introduction	1
1.1 Background	1
1.2 Problem Overview	4
1.3 Plan of Attack	11
1.4 Overview	12
II. Truth Model	14
2.1 Introduction	14
2.2 Target Model	14
2.3 Trajectory Description	21
2.4 Measurement Model	29
2.5 Projection of Multiple Hot Spots	36
III. Kalman Filter	49
3.1 Introduction	49
3.2 Acceleration and Atmospheric Models	50
3.3 State Space Model	51
3.4 State Propagation	55
3.5 Measurement Update Equation	59
3.6 \hat{Q}_D Estimation	60
IV. Methods of Correlation	64
4.1 Introduction	64
4.2 Image Resolution	65
4.3 Correlation Methods	66
4.4 Maximum Correlation Detection	77
4.5 Analysis of Correlation Methods	83

Contents

	Page
V. Performance Analysis	98
5.1 Introduction	98
5.2 Tracking Ability	98
5.3 Variation of Truth Model Parameters	100
5.4 Variation of Data Processing Parameters	101
5.5 Variation of Filter Parameters	102
5.6 Plotting Results	104
5.7 Analysis of Filter Performance	115
5.8 Evaluation of Correlation Methods	117
5.9 Evaluation of Harnly and Jensen EKF	118
5.10 Evaluations Using Trajectory 1	134
5.11 Evaluation for Rolling Maneuvers.	136
5.12 Evaluation for a Two-G Pullup Maneuver.	139
5.13 Evaluation for a Five-G Pullup Maneuver	141
5.14 Adaptive Q_{fd} Estimation Evaluation	143
5.15 Maneuvers Out of the x-y Plane	145
5.16 Summary of Test Cases	148
VI. Conclusions and Recommendations	150
6.1 Introduction	150
6.2 Conclusions	150
6.3 Recommendations	152
Bibliography	155
Appendix A: Intensity Centroid Projection Model for Multiple Hot Spots	157
Appendix B: Derivation of Q_{fd} for the Kalman Filter, Chapter III	164
Appendix C: Plots of Tracking Errors	169
Appendix D: Computer Software (Correlator/Kalman Filter)	284
Appendix E: Computer Software (Trajectory Model)	351
Vita	357

List of Figures

Figure		Page
1	Data Processing Algorithm	5
2	Research Plan of Attack Flow Diagram . . .	11
3	Inertial Coordinate Frame.	19
4	Azimuth Geometry	19
5	Elevation Geometry	20
6	Trajectory 1	23
7	Trajectory 2	24
8	Trajectory 4	27
9	Out-of-Plane Coordinate Frame Rotations. .	28
10	Image Intensity Characteristics	30
11	Image Projection	32
12	Semimajor Axis Projection	33
13	$\vec{e}_\beta - \vec{e}_{z\alpha}$ Plane	38
14	$\vec{e}_\beta - \vec{e}_{z\alpha}$ Plane Translation	38
15	$\vec{e}_\beta - \vec{e}_{z\alpha}$ and H-Frame Unit Vectors	39
16	Initial Ellipsoidal Centers	40
17	Roll Maneuver Geometry	41
18	Hot Spot Offsets	42
19	Distance Translated in $\vec{e}_\beta - \vec{e}_{z\alpha}$ Plane . . .	43
20	π Radians Roll	47
21	2π Radians Roll	48
22	Functions to be Correlated	67
23	Arrays to be Correlated	69
24	FLIR and Template Arrays	70

List of Figures

Figure		Page
25	Input-Output Relationships	72
26	FFT Correlation (Threshold = .3)	79
27	FFT Correlation (Threshold = .5)	80
28	Thresholding (FFT Correlation Method)	81
29	Thresholding (Phase Correlation Method)	82
30	Histogram of Errors - Direct Method (2-Level Quantization, 1 Hot Spot)	86
31	Histogram of Errors - Direct Method (6-Level Quantization, 1 Hot Spot)	87
32	Histogram of Errors - FFT Method (Threshold = .3, 1 Hot Spot)	88
33	Histogram of Errors - Phase Method (Threshold = .7, 1 Hot Spot)	89
34	Histogram of Errors - Direct Method (2-Level Quantization, 3 Hot Spots)	91
35	Histogram of Errors - Direct Method (6-Level Quantization, 3 Hot Spots)	92
36	Histogram of Errors - FFT Method (Threshold = .3, 3 Hot Spots)	93
37	Histogram of Errors - Phase Method (Threshold = .7, 3 Hot Spots)	94
38(a-j)	Case 1	105-114
39(a-d)	Case A	119-122
40(a-d)	Case B	125-128
41(a-d)	Case C	130-133
A-1	Project on the FLIR Plane	163
(C-2)-(C-20)	Case 2-Case 20	170-283

List of Tables

Table		Page
I	Correlation Errors (Single Hot Spot Target)	85
II	Correlation Errors (Three Hot Spot Target)	90
III	Correlator Performance Results	116
IV	Harnly-Jensen EKF Performance	123
V	Trajectory 1 Evaluations	135
VI	Roll Maneuver Elevations	138
VII	Trajectory 2 (Two-g Pullup) Evaluations . .	140
VIII	Trajectory 2 (Five-g Pullup) Evaluations .	142
IX	Q_{fd} Estimation Evaluation	144
X	Out of Plane Maneuvers	147
XI	Summary of Test Cases	149

List of Symbols

Symbol

t	time
\underline{u}	deterministic velocity input function
\underline{v}	velocity vector
$v_{\perp LOS}$	target velocity component perpendicular to the FLIR image plane line of sight
$\alpha(t)$	azimuth
$\beta(t)$	elevation
γ	angle between velocity vector and image plane
ω	roll rate
ϕ	roll angle
δ	distance from aircraft center of mass to intensity function centroid
σ_g	dispersion of Gaussian intensity function
w	white noise
x_{peak}	horizontal coordinate of Gaussian intensity function maximum
y_{peak}	vertical coordinate of Gaussian intensity function maximum
\underline{x}	state vector
\underline{z}	measurement vector
I_{max}	maximum intensity received from target
r	range
r_h	horizontal range
T_{df}	correlation time assumed for target acceleration
T_{af}	correlation time assumed for atmospheric jitter

List of Symbols

Symbol

σ_{df}^2	assumed target acceleration noise variance
σ_{af}^2	assumed target acceleration atmospheric jitter variance
$\hat{x}_f(t_i^-)$	propagated filter state estimate vector before measurement incorporation at time t_i
$\hat{x}_f(t_i^+)$	filter state estimate vector after measurement incorporation at time t_i
$P(t_i^-)$	conditional filter state covariance matrix propagated from time t_{i-1} to time t_i
Φ_f	filter state transition matrix
F_f	filter plant matrix
G_f	filter noise distribution matrix
H_f	linear combination of the filter state variables which contribute to the respective measurement elements
v_f	additive noise corruption
Q_f	noise covariance kernel descriptor
$c\sqrt{}$	Cholesky square root

subscripts

a	atmospherics
d	dynamics
f	filter
t	true
B	body
H	horizontal

Abstract

Over the past four years considerable work has been accomplished at the Air Force Institute of Technology to improve the tracking capability of the high energy laser weapon against airborne targets. In this research, many of the prior concepts are incorporated into a correlator/Kalman filter to develop a tracker capable of providing precise target position estimates in a dynamic short-range environment using a Forward Looking Infrared sensor (FLIR) to provide measurement data. Digital signal processing is employed on the FLIR data to identify the underlying target intensity shape function when the target under consideration has either single or multiple hot spots. The estimated target shape function is then used as the template in a correlation algorithm, where spatial and frequency domain correlation techniques were explored, to determine the offsets between the template and the incoming measurement. These offsets are used as "pseudomeasurements" in a linear Kalman filter which exploits knowledge of the process dynamics and statistical knowledge of the correlator errors to enhance the position estimates.

ENHANCED TRACKING OF AIRBORNE TARGETS USING A CORRELATOR/KALMAN FILTER

I. Introduction

The rapid growth in laser technology since the early 1960's has led researchers in a wide variety of disciplines to investigate possible applications of this device. The unique characteristics of the laser make it a highly desirable weapon system, with one prospective military application involving the use of laser energy to destroy airborne targets (1:14-17, 2:16-19). Since the directed energy is transmitted at the speed of light from the source to its destination, the energy arrives at the target almost instantaneously, eliminating the need for lead computation. Additionally, the laser can potentially deposit large amounts of energy on a target in a short period of time, thereby destroying or disabling the target without firing an expensive missile for each engagement opportunity as in conventional systems.

In actual implementation, this system requires a high energy laser, a very accurate pointing system, and a very accurate estimate of the target position. The precise pointing control and target position estimates are required to concentrate the directed energy on a specific part of the target, instead of "painting" the entire target with energy, and to maintain the laser beam on the target long enough to disable the target.

1.1 Background

This study is a continuation of other research projects conducted at the Air Force Institute of Technology

(AFIT) over the past four years which have investigated possible solutions to the target tracking problems associated with directed energy weapons. Currently, the Air Force Weapons Laboratory (AFWL) uses correlation trackers to provide precise target position estimates to feedback controllers in the presence of disturbances. These disturbances include any effect which can cause relative motion between the beam and the target, such as true target motion, atmospheric jitter, and sensor measurement errors (3:2). A correlation tracker compares new target information received from a sensor with a template, consisting of either predetermined or previous real-time data. Correlation techniques are then used to estimate the relative position offsets from one data frame to the next. This relative position information is then used to drive the tracking servo error voltages in azimuth and elevation to keep the target image centered within the sensor's field-of-view (FOV). Although various sensors capable of providing target position information are available, the one currently of the most interest due to its passive nature, and the one which will be used in this research, is the Forward Looking Infra-Red sensor (FLIR) (4).

The correlation algorithm is well suited to many practical applications because this method requires no a priori information. However, in many tracking situations certain target parameters such as shape, size, and acceleration characteristics are either known or could be estimated, which would enhance the tracker's ability to estimate the true target position. The effects of atmospheric disturbances on radiated waveforms are known and statistical data could be used to separate the true target motion from apparent motion due to disturbances. This separation is crucial since the directed energy beam

does not undergo the same distortion as the infrared wavefront emanating from the target. Additionally, statistical data on FLIR noise and background noise is available, and can be used to provide a better estimate of the target's true position (5:222). The desire to exploit this knowledge, unused by the correlation trackers, led researchers at AFIT to investigate the possibility of designing an extended Kalman filter as an alternative to correlation trackers. This method was selected because the reduced computational loading incurred with implementing this filter, when compared to other nonlinear filters is great enough to warrant an evaluation of its performance.

An extended Kalman filter can incorporate estimates of target parameters such as size, shape, and acceleration, as well as statistical information on the effects of atmospheric distortion on radiated wavefronts, and sensor errors, to aid in separating the true target motion from the apparent motion created by disturbances. In initial research efforts, the extended Kalman filter outperformed the correlation tracker against targets exhibiting a single point source of infrared radiation, or "hot spot", when the intensity function was relatively well-known by the filter, and the internal filter structure was designed to depict the tracking environment to be encountered (5,6). The performance enhancement achieved by the extended Kalman filter under these controlled conditions led to further efforts to design a filter capable of accurate tracking in an environment when the target intensity pattern on the FLIR image plane is not well known a priori (7,8). Additionally, as in realistic situations, the tracker has to be capable of tracking targets exhibiting both single and multiple hot spots which change in time, so the target intensity pattern must be identified in real time.

In the thesis by Captain S.K. Rogers (8), a digital processing algorithm was developed which is capable of identifying the target intensity pattern in real time for a single or multiple hot spot target when neither the functional form of the target intensity nor the number of hot spots is known a priori. This algorithm, shown in the upper path of Figure 1 and detailed in section 1.2, was implemented by Rogers in two trackers. The first tracker used the estimated target shape intensity directly in the measurement model of an extended Kalman filter while the second tracker used the estimated target shape as a template in an enhanced correlator to provide target measurement information to a linear Kalman filter. A linear Kalman filter can be utilized in this tracker because the outputs of the correlation algorithm are offset distances which are linear functions of the chosen state variables (see Chapter 3). Both trackers exhibited significant performance potential against targets having benign dynamics, with the extended filter having larger mean errors and the correlator filter having larger standard deviations (8,9). However, the reduced computational burden associated with the actual implementation of the correlator/Kalman filter, and the enhanced potential of using this filter with optical processing alternatives, strongly urges further investigation of this approach (9).

1.2 Problem Overview

The purpose of this research is to expand on the Rogers' work by investigating the feasibility of utilizing a linear Kalman filter in cascade with a correlation algorithm, using either the correlation method employed by Rogers or an alternative method as developed in Chapter 4. Alternative correlation methods were explored to determine if either a reduction in the

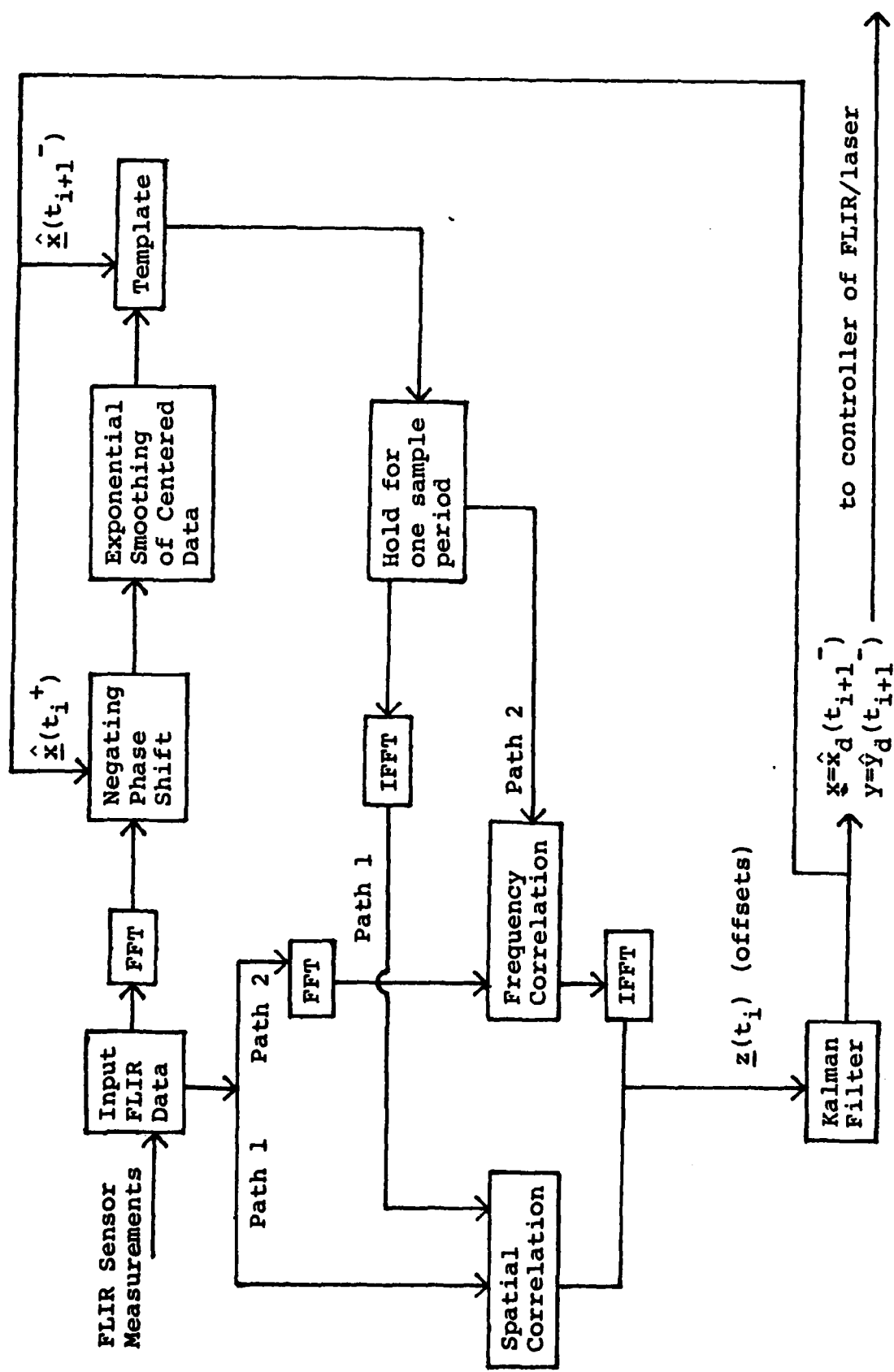


Figure 1. Data Processing Algorithm

computational loading of the correlator could be achieved without a degradation in performance, or another method is available which reduces the mean errors and standard deviations of the correlator without a substantial computational increase. The ability of the correlator/Kalman filter to track targets exhibiting single and multiple hot spots over a varying range of dynamic profiles will be evaluated. These dynamic profiles were specifically designed to evaluate the ability of the filter to track changes in the target dynamics as well as changes in the target's intensity pattern. The target scenarios and reasons for selection of those scenarios are detailed in Chapter 2. In the single hot spot case, the performance achieved by the extended Kalman filter employed by Harnly and Jensen (6) will serve as the standard against which the correlator/Kalman filter performance will be evaluated.

As shown in Figure 1., measurement information of target position for the correlator/Kalman filter is generated by the FLIR sensor. In this research the FLIR tracking window is 8 pixels wide by 8 pixels high where a pixel length of 20μ radians is used, although an expansion of this window is possible. The measurement information presented to the tracking algorithm is the average intensity over each of the 64 square pixel elements within the 8 x 8 tracking window. The concept of the correlator/Kalman filter is to generate an accurate reference image of the target's intensity function on the FLIR image plane to serve as a template for correlation with new data received from the sensor. In Figure 1., either path 1 or path 2 is followed depending on the correlation method being used. The results of the correlation of these two data arrays namely, the indicated angular position offsets of the target centroid, Equation (1-3), from the center of the FLIR field-of-view

(FOV), can then be used in the measurement model portion of a linear Kalman filter to estimate target offsets from the center of the FOV. The linear Kalman filter processes the measurement vector, $\underline{z}(t_i)$, using

$$\underline{x}(t_i^+) = \underline{x}(t_i^-) + \underline{K}(t_i) \{ \underline{z}(t_i) - \underline{H} \underline{x}(t_i^-) \} \quad (1-1)$$

where

$\underline{x}(t_i^+)$ = state estimate vector after measurement incorporation at time t_i

$\underline{x}(t_i^-)$ = state estimate vector propagated from previous measurement update to time t_i

$\underline{K}(t_i)$ = Kalman filter gain

$\underline{z}(t_i)$ = measurement vector of the estimated offset distance in the horizontal and vertical direction from the center of the FLIR plane to the centroid of the intensity function

$\underline{H}(t_i)$ = linear combination of the states which contribute to the respective measurements

A detailed development of the Kalman filter equations is given in Chapter 3. These estimated offsets are to be regulated to zero by using the estimates as inputs to a pointing controller that points the laser beam and the center of the FLIR FOV appropriately. The state estimates are also used to aid in estimating the shape function in the data processing algorithm.

The primary focus of the Rogers thesis was to generate an accurate estimate of the target's intensity shape function and to evaluate its performance in a benign tracking environment where the target did not leave the FLIR FOV in one sample period. Thus, a four-state estimate vector, $\underline{x}(t_i)$, consisting of estimates of the x and y positions due to true target dynamics and the x and y positions due to atmospheric effects was utilized (10). The position estimates are along the FLIR horizontal and vertical axes respectively. However, in order to track targets over a wide dynamic range the capability to

to predict future target position is required; thus, in this research the four-state estimate vector is replaced by an eight-state estimate vector, consisting of estimates of the target's position due to dynamics and atmospheric in the x and y directions as well as estimates of the target's velocity and acceleration in both the x and y directions.

In the lower path of Figure 1, the Kalman filter incorporates the measurement information, using Equation (1-1) at time t_i , and utilizes its internal dynamics model to propagate its estimate of where the true target position will be at the next sample period, t_{i+1} , using

$$\underline{x}(t_{i+1}^-) = \underline{\phi}(t_{i+1}, t_i) \underline{x}(t_i^+) \quad (1-2)$$

where

$\underline{\phi}(t_{i+1}, t_i)$ = filter state transition matrix as defined in Equation (3-11)

The FLIR is located so as to zero out the estimated target dynamics position components which are the first two states of $\underline{x}(t_{i+1}^-)$. The atmospheric disturbances which can cause apparent translational offsets of the intensity function on the FLIR image plane are also accounted for by $\underline{x}(t_{i+1}^-)$ and the correlator template is positioned with this estimate. The incoming measurement array is then correlated with the template to determine the position offset between the two arrays. Errors in the correlation algorithm are reduced by processing the position offset estimates with the Kalman filter which exploits statistical knowledge of the correlator errors to produce a better position estimate for actual tracking purposes (9:13-14). The Kalman filter then uses its internal dynamics model to propagate its best estimate of the states at the next sample time, $\underline{x}(t_{i+2})$.

The upper path of Figure 1 is designed to generate the template for correlation with the incoming FLIR

data array. The fundamental concept of the algorithm is to utilize the fact that the actual target image will change rather slowly relative to a given sample period while background noises will typically change more rapidly. This path generates an estimated representation of the average value of the target intensity pattern over each pixel, which would be observed if the measurements were noise-free and the filter state estimates were perfect. The location of the centroid of the target intensity pattern is the sum of the effects caused by true target dynamics plus apparent translational motion caused by atmospheric disturbances. In the x-direction the centroid of the target's intensity profile is defined by

$$x_{\text{centroid}} = x_{\text{dynamics}} + x_{\text{atmospherics}} \quad (1-3)$$

and control action is applied to zero out the estimated x_{dynamics} , and similarly in the y-direction. Thus, under these conditions the target's intensity profile will be offset from the center of the FLIR FOV by the predicted atmospheric states, $x_a(t_i^-)$ and $y_a(t_i^-)$.

To generate the estimated target intensity pattern from the noise-corrupted FLIR data, interframe smoothing is employed to attenuate the noise. The raw FLIR data is put through an FFT to allow for efficient data processing and possible spatial frequency filtering. In order to center the target in the original spatial domain, the appropriate negating phase shift is applied to this transformed image. The offset estimate for the frame at t_i is provided by $x_{\text{centroid}}(t_i^+)$ and $y_{\text{centroid}}(t_i^+)$ based on Equation (1-3) as obtained from the Kalman filter. The phase shift is computed according to the shifting property of the two-dimensional discrete Fourier transform (8:Equation (2-7)), with the output of the "Negating Phase Shift" appearing as the result

of passing a centered target image through a FFT.

This result can then be averaged with the most recent N such centered and transformed data frames to attenuate the background noise and accenuate the underlying target pattern. Instead of explicitly storing N data sets, finite-memory averaging is approximated by using exponential smoothing

$$\hat{G}(t_i) = \alpha G(t_i) + (1-\alpha)\hat{G}(t_{i-1}) \quad (1-4)$$

where

$G(t_i)$ = current data frame value of G

$\hat{G}(t_i)$ = current estimate of G

$\hat{G}(t_{i-1})$ = previous smoothed estimates of G

α = smoothing parameter; $0 < \alpha < 1$

Thus, a smaller α corresponds to a longer finite memory being approximated as appropriate for slowly changing target patterns (9:8-11).

At time t_i , the output of the "Exponential Smoothing of Centered Data" block is a representation of the FFT of the estimated target intensity pattern corresponding to a centered image. This pattern is then evaluated using $\underline{x}(t_{i+1}^-)$ which, due to the previous controller action, corresponds to offsetting this FFT by the phase shift corresponding to the atmospheric state components of $\underline{x}(t_{i+1}^-)$ as generated by the Kalman filter. The intensity function, $h(\underline{x}(t_{i+1}^-), t_{i+1})$, is then ready for use as the template in the correlation process at the next sample time t_{i+1} , and is placed in a one period storage location awaiting the next measurement. Note the previous contents of this location which corresponds to the estimated intensity function generated by the smoothing algorithm at t_{i-1} , was used as the template at t_i , i.e. $h(\underline{x}(t_i^-), t_i)$, to generate $\underline{x}(t_i^+)$ and $\underline{x}(t_{i+1}^-)$. Depending on the correlation method used, the correlation

of the template and the incoming data array is either accomplished in the frequency or spatial domain. For a detailed development of the upper path of Figure 1, see reference 8.

1.3 Plan of Attack

This section presents a general overview of the flow of this research as depicted in Figure 2.

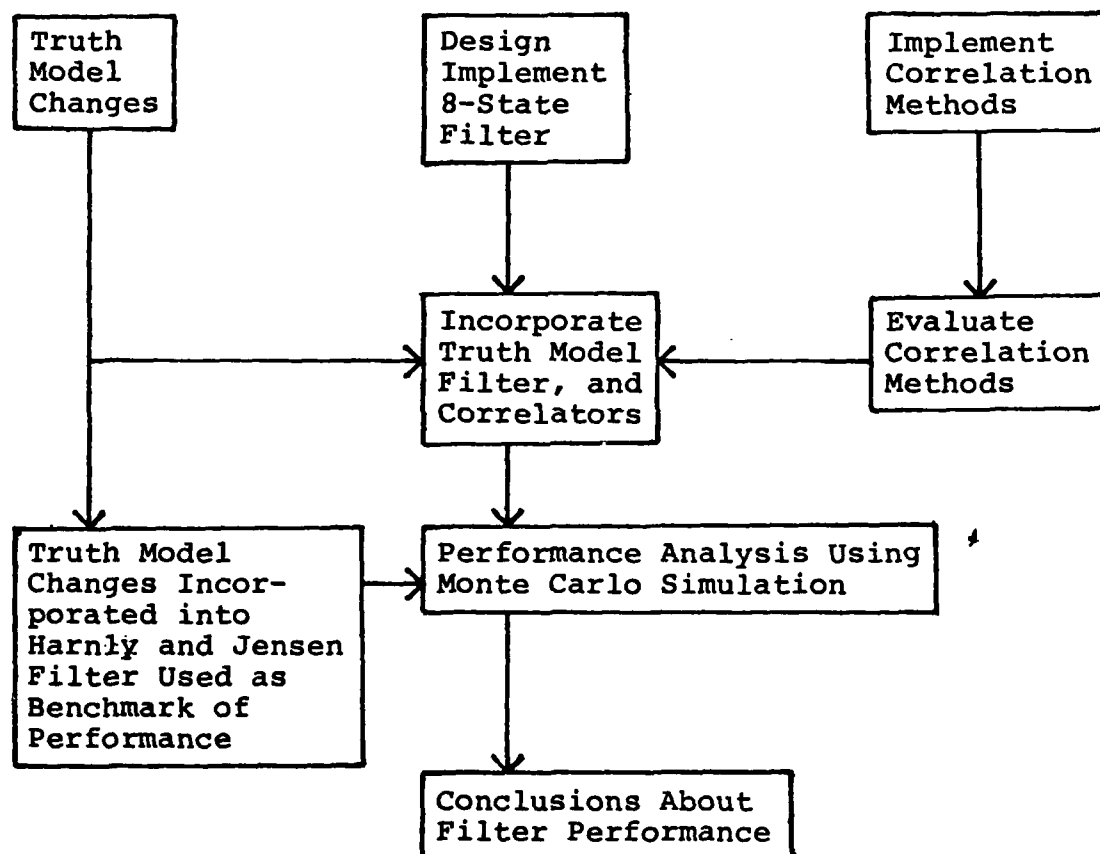


Figure 2. Research Plan of Attack Flow Diagram

The computer simulation developed by Captain Rogers (8) was used as the basic computer program to which changes were made to evaluate the performance of the correlator/Kalman filter in realistic tracking environments. This approach required that realistic target trajectories be incorporated into the truth model in place of the benign target trajectory model used by Rogers. Additionally, a model was developed to account for the movement of both single and multiple hot spots relative to the aircraft center of mass and to project the hot spot or spots onto the FLIR image plane for measurement data. The four-state Kalman filter used by Rogers was replaced by the eight-state Kalman filter which is developed in Chapter 3. Finally, based on a literature search of available correlation methods, alternative correlation methods were selected based on the criteria previously discussed, for evaluation against the correlation method used by Rogers, see Chapter 4. As a result of this evaluation, one alternative method, along with the method developed by Rogers, was implemented in cascade with the Kalman filter.

The truth model, Kalman filter, and correlator models were incorporated into a computer simulation for a performance analysis using a Monte Carlo simulation. Additionally, the truth model changes were incorporated into an extended Kalman filter developed by Harnly and Jensen (6) to track only single hot spot targets with intensity functions well described as bivariate Gaussian to provide a benchmark of performance for the proposed algorithm.

1.4 Overview

Chapters II, III, and IV, discuss in detail the mathematical models used in the computer simulation. Chapter II presents the truth model which represents

the environment from which measurements are taken. Chapter III describes the eight-state linear Kalman filter used in the computer simulation. Chapter IV presents a mathematical development of the methods of correlation used and an analysis of the performance of these correlation methods. Chapter V presents a performance analysis of the linear correlator/Kalman filter and the Harnly and Jensen filter against targets exhibiting a wide range of dynamic profiles and both single and multiple hot spot intensity shapes on the FLIR image plane. Chapter VI presents the conclusions and recommendations.

II. Truth Model

2.1 Introduction

The truth model is the best mathematical representation of the real world process to be simulated that is available to, and can be implemented by, the researcher. In this study, the truth model portrays the motion of the target, an air-to-air missile or a multiple engine aircraft, in inertial space and the intensity function emitted by the target which is distorted by atmospheric disturbances. The resulting infrared image is then projected onto the two-dimensional FLIR image plane. With this distorted target intensity pattern projected onto the FLIR image plane, spatially correlated and temporally uncorrelated noise which accounts for background and inherent FLIR noises, are added to create the corrupted measurement array, $\underline{z}(t_i)$, for incorporation by the correlator/Kalman filter.

This chapter outlines the truth model used in this study. For details of how the components of the model were developed, consult the cited references.

2.2 Target Model

This section develops the equations which represent the trajectory of the target being tracked in inertial space and translates that into motion on the two-dimensional FLIR image plane, which is represented by movement of the target intensity pattern within the FLIR field-of-view. Although this motion consists of several components, in this research the apparent target motion due to boresight errors, FLIR system vibrations, etc., are assumed to be negligible compared to the motion due to target dynamics and atmospheric jitter. Thus, the continuous time target model describes the apparent target motion due to actual target dynamics and atmospheric

by means of differential equations, as well as stochastically to represent the statistical properties of the physical processes of concern and to account for the unmodeled physical effects.

As developed by Harnly and Jensen (6), the continuous target dynamics model accounts for the true location of the target center of mass on the two-dimensional FLIR image plane in the horizontal direction, α , and in the vertical direction, β . A deterministic model was used to provide a time history of the target location so that specific trajectories could be generated for tracker evaluation although a stochastic model could be used. Thus,

$$\dot{x}_1(t) = \dot{\alpha}(t) = \text{horizontal velocity}$$

$$\dot{x}_2(t) = \dot{\beta}(t) = \text{elevation velocity}$$

or

$$\dot{\underline{x}}_D(t) = \underline{u}(t) = [\dot{\alpha}(t) \dot{\beta}(t)]^T \quad (2-1)$$

While a target trajectory could also have been generated by simply reading in time histories of α and β , a deterministic model of the form of Equation (2-1) was selected so a switch from the deterministic form to either a first-order Gauss-Markov process, a Brownian motion stochastic process, or the sum of a deterministic and stochastic model could readily be implemented if desired. The atmospheric disturbances, as developed by Mercier (10), were modeled as third-order Gauss-Markov processes, described by the output of a shaping filter with a frequency domain transfer function of

$$\frac{x_A}{w_3} = \frac{K(14.14)(659.5)^2}{(s+14.14)(s+659.5)^2} \quad (2-2)$$

driven by a unit strength white Gaussian noise w_3 . A duplicate independent model is used to generate y_A

in the vertical FLIR plane direction. Physically, this corresponds to jitter in each of the two FLIR image plane coordinate directions being modeled as the output of a linear third order system driven by white Gaussian noise, developed to match the power spectral density characteristics of the observed physical jitter phenomenon. The atmospheric jitter is then represented in both FLIR plane directions by a stochastic differential equation of the form

$$\dot{\underline{x}}_A(t) = \underline{F}_A(t)\underline{x}_A(t) + \underline{G}_A(t)\underline{w}_A(t) \quad (2-3)$$

where

$\underline{x}_A(t)$ = six atmospheric noise states

$\underline{F}_A(t)$ = atmospheric plant matrix

$\underline{G}_A(t)$ = atmospheric noise distribution matrix

$\underline{w}_A(t)$ = two-dimensional vector of white Gaussian noise inputs with statistics:

$$E\{\underline{w}_A(t)\} = \underline{0}$$

and

$$E\{\underline{w}_A(t)\underline{w}_A^T(t+\tau)\} = \underline{Q}_A(t)\delta(\tau)$$

After augmenting the atmospheric states to the dynamic states, Equations (2-1) and (2-3), and writing the equations in an equivalent discrete time form (6), the solution to the discretized truth model propagation of the motion of the target intensity function has the form

$$\underline{x}(t_{i+1}) = \underline{\phi}(t_{i+1}, t_i)\underline{x}(t_i) + \begin{bmatrix} \underline{B}_d(t_i) \\ \underline{0} \end{bmatrix} \underline{u}_d(t_i) + \begin{bmatrix} \underline{0} \\ \underline{c} \sqrt{\underline{Q}_{Ad}} \end{bmatrix} \underline{w}_{Ad}(t_i) \quad (2-4)$$

where

$\underline{x}(t_i)$ = state vector of the two dynamic states and six atmospheric states

$\underline{B}_d(t_i)$ = input matrix for dynamics = $\int_{t_i}^{t_{i+1}} \underline{\phi}(t_{i+1}, t_i) \underline{B}(\tau) d\tau$

$\underline{u}_d(t_i)$ = piecewise constant function (constant between sample times) evaluated at the interval midpoint as an approximation to the integral of $\alpha(t)$ and $\beta(t)$ from t_i to t_{i+1} . Explicitly,

$$\underline{u}_d(t_i) = \begin{bmatrix} \dot{\alpha}(t_i + \frac{\Delta t}{2}) \\ \dot{\beta}(t_i + \frac{\Delta t}{2}) \end{bmatrix}$$

$$\Delta t = t_{i+1} - t_i$$

$\underline{w}_{Ad}(t_i)$ = discrete-time white Gaussian noise with statistics:

$$E\{\underline{w}_{Ad}(t_i)\} = \underline{0}$$

$$E\{\underline{w}_{Ad}(t_i) \underline{w}_{Ad}^T(t_j)\} = \underline{I} \delta_{ij}$$

and

$$\begin{aligned} \underline{c}\sqrt{\underline{Q}_{Ad}(t_i)} \underline{c}\sqrt{\underline{Q}_{Ad}(t_i)}^T &= \underline{Q}_{Ad}(t_i) = \\ &= \int_{t_i}^{t_{i+1}} \underline{\phi}_A(t_{i+1}, \tau) \underline{G}_A(\tau) \underline{Q}_A(\tau) \underline{G}_A^T(\tau) \underline{\phi}_A^T(t_{i+1}, \tau) d\tau \end{aligned}$$

where $\underline{c}\sqrt{\underline{Q}_{Ad}}$ represents the Cholesky square root of \underline{Q}_{Ad} . See reference (12) for development.

Thus,

$$E\{\sqrt{Q_{Ad}} w_{Ad}(t_i)\} = 0$$

$$E\{\sqrt{Q_{Ad}} w_{Ad}(t_i) w_{Ad}^T(t_j) \sqrt{Q_{Ad}}^T\} = Q_{Ad} \delta_{ij}$$

Explicitly, the state transition matrix is

$$\underline{\Phi}(t_{i+1}, t_i) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & e^{-A\Delta t} & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & e^{-B\Delta t} \Delta t e^{-B\Delta t} & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & e^{-B\Delta t} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & e^{-A\Delta t} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & e^{-B\Delta t} \Delta t e^{-B\Delta t} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-B\Delta t} \end{bmatrix}$$

The three by three submatrices in $\underline{\Phi}(t_{i+1}, t_i)$ are identical and propagate the atmospheric disturbances in azimuth and elevation with poles A and B as shown in Equation (2-2).

$$\underline{B}_d(t_i) = \begin{bmatrix} \Delta t & 0 \\ 0 & \Delta t \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \\ 0 & 0 \end{bmatrix}$$

For a detailed development of Equation (2-4) consult ref. 6.

The last step in the target dynamics model is to define the azimuth velocity, $\dot{\alpha}(t)$, and elevation velocity, $\dot{\beta}(t)$, in the FLIR image plane. To simulate this motion, the target velocity was initially calculated in an inertial frame, with its origin at the center of the FLIR image plane, and then projected onto the two-dimensional FLIR image plane. The relationship between the FLIR frame and inertial frame is shown in Figure 3.

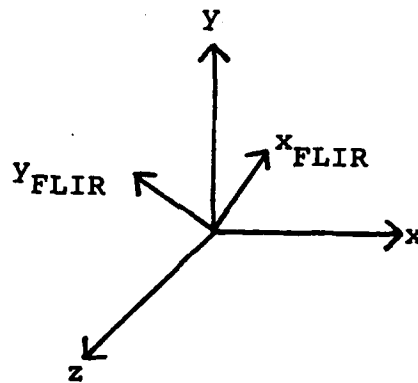


Figure 3. Inertial Coordinate Frame

The geometry involved in the azimuth direction is as shown in Figure 4.

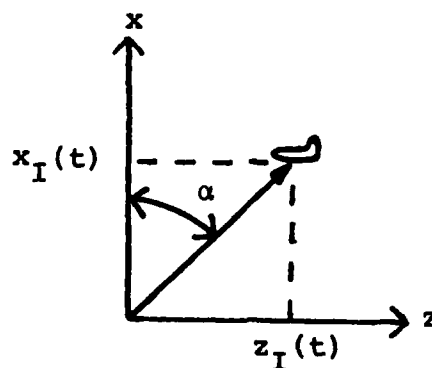


Figure 4. Azimuth Geometry

From Figure 4,

$$\alpha(t) = \tan^{-1} \frac{z_I(t)}{x_I(t)} \quad (2-5)$$

and

$$\dot{\alpha}(t) = \frac{x_I(t)\dot{z}_I(t) - z_I(t)\dot{x}_I(t)}{z_I^2(t) + x_I^2(t)} \quad (2-6)$$

Equation (2-6) yields an azimuth velocity in radians/sec which is converted to FLIR image plane units, pixels/sec, by dividing $\dot{\alpha}(t)$ by 20×10^{-6} radians/pixel (6:33).

Similarly, Figure 5 shows the geometry involved in computing the elevation velocity.

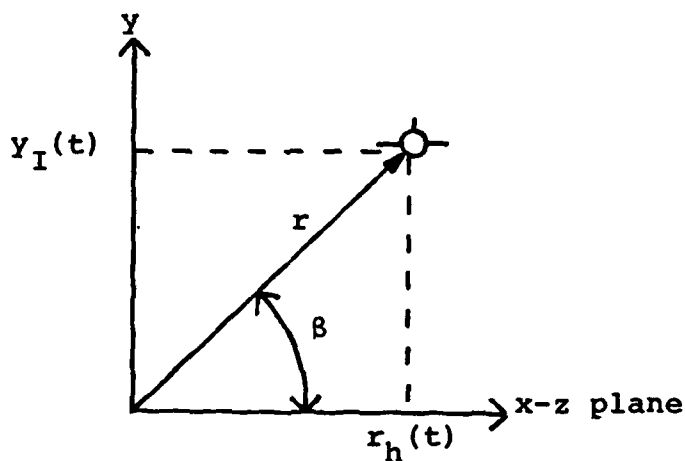


Figure 5. Elevation Geometry

where

$$r = \text{range} = \{x_I^2(t) + y_I^2(t) + z_I^2(t)\}^{1/2}$$

$$r_h = \text{horizontal range} = \{x_I^2(t) + z_I^2(t)\}^{1/2}$$

From Figure 5, the tangent function can be used to find β :

$$\beta = \tan^{-1} \frac{y(t)}{r_h(t)} \quad (2-7)$$

and

$$\begin{aligned} \dot{\beta}(t) &= \frac{r_h(t)\dot{y}_I(t) - y_I(t)\dot{r}_h(t)}{r_h^2(t) + y_I^2(t)} \\ &= \frac{r_h(t)\dot{y}_I(t) - y_I(t) \left[\frac{x_I(t)\dot{x}_I(t) + z_I(t)\dot{z}_I(t)}{r_h(t)} \right]}{r^2(t)} \end{aligned} \quad (2-8)$$

The elevation velocity is then converted to pixels/sec also.

By inserting Equations (2-6) and (2-8) into Equation (2-4), deterministic time histories for the truth model propagation can be generated to produce the desired trajectories for the target.

2.3 Trajectory Description

A thorough evaluation of the filter's tracking performance requires that specific trajectories be designed to test the ability of the filter to maintain an accurate estimate of the target's true position under various circumstances. A benign crossing trajectory was selected to serve as the baseline for the filter's performance, as this should be an easy trajectory for the filter to track. For the multiple hot spot case, a rolling trajectory was designed to evaluate how well the data

processing algorithm can reconstruct the target intensity pattern when the target's intensity pattern exhibits changes on the FLIR plane relative to the aircraft center of mass. Several trajectories were designed to evaluate the filter's ability to maintain track on a target performing highly dynamic maneuvers. Note, as will be discussed later, under these circumstances while the center of the target intensity pattern relative to the aircraft center of mass does not change, the orientation of the elliptical intensity pattern as projected onto the FLIR image plane will change. Finally, a trajectory was designed to evaluate the ability of the algorithm to estimate the target position when the target is performing a high-g maneuver and the dynamic shape of the intensity pattern is also changing simultaneously.

The benign trajectory which served as the baseline for filter performance was simulated by having the target fly a cross-range trajectory parallel to the x-y plane. The flight path referred to as trajectory 1, is depicted in Figure 6.

(NOT USED)

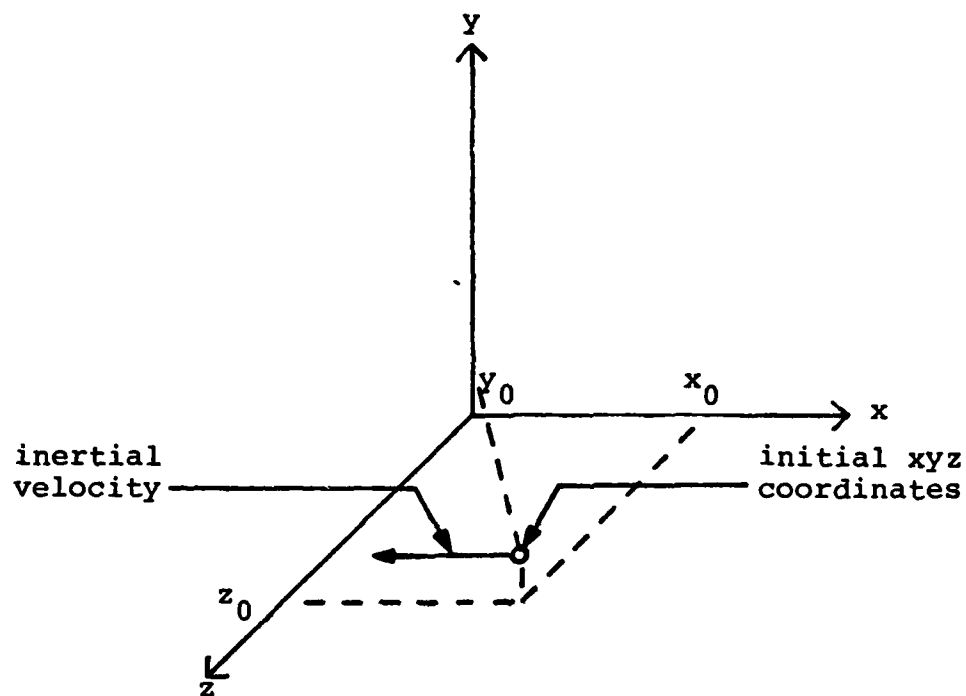


Figure 6. Trajectory 1

The inertial location of the target at t_0 is

$$x_I(t_0) = 5000.0 \text{ m}$$

$$y_I(t_0) = 500.0 \text{ m}$$

$$z_I(t_0) = 20000.0 \text{ m}$$

The trajectory throughout the simulation is given by

$$\dot{x}_I(t) = -1000. \text{ m/sec}$$

$$\dot{y}_I(t) = 0. \text{ m/sec}$$

$$\dot{z}_I(t) = 0. \text{ m/sec}$$

Trajectory 1 was used to evaluate the performance of the filter against a multiple hot spot target flying either a wings level trajectory or a roll maneuver. Roll rates of .5 radians/sec and 1.0 radians/sec were used as being representative of realistic performance.

To evaluate the performance of the filter against a target performing a maneuver, the target was initialized with the same inertial location and velocity as shown in Figure 6. Two seconds into the simulation, a constant g pullup maneuver was initiated which lasted for three seconds, and the simulation was terminated. The flight path, trajectory 2, is depicted in Figure 7.

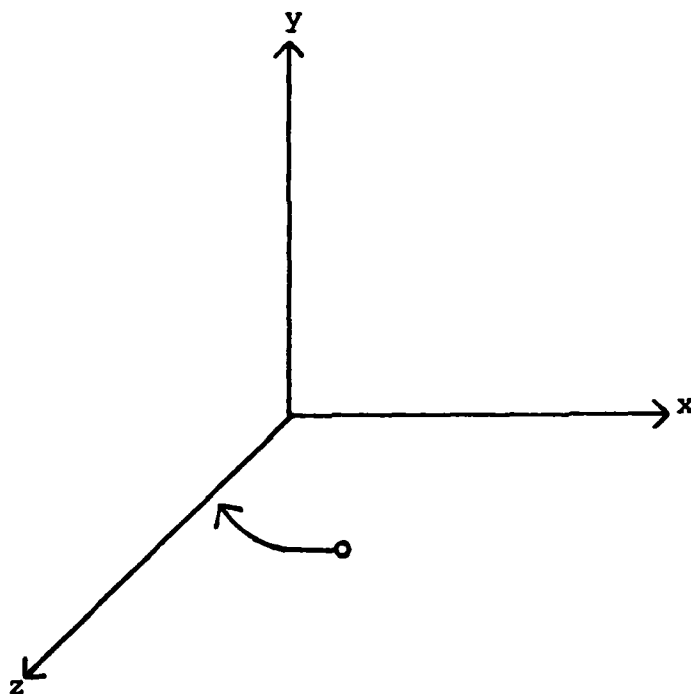


Figure 7. Trajectory 2

The trajectory prior to the pull-up maneuver is described by

$$\dot{x}_I(t) = -1000 \text{ m/sec}$$

$$\dot{y}_I(t) = 0.$$

$$\dot{z}_I(t) = 0.$$

and is initiated at inertial coordinates

$$x_I(t_0) = 5000 \text{ m}$$

$$y_I(t_0) = 500 \text{ m}$$

$$z_I(t_0) = 20000 \text{ m}$$

At $t_i=2.0$ sec, the pull-up maneuver is initiated at the inertial coordinates

$$x_I(t_2) = 3000 \text{ m}$$

$$y_I(t_2) = 500 \text{ m}$$

$$z_I(t_2) = 20000 \text{ m}$$

The velocity equations are then

$$\dot{x}_I(t) = -1000 \cos\{\omega(t-2)\} \text{ m/sec}$$

$$\dot{y}_I(t) = 1000 \sin\{\omega(t-2)\} \text{ m/sec}$$

$$\dot{z}_I(t) = 0. \text{ m/sec}$$

and the inertial position of the target can be determined by

$$x_I(t) = x_I(t_0) - 2000 - \{1000/\omega\} \sin\{\omega(t-2)\} \text{ m}$$

$$y_I(t) = y_I(t_0) + \{1000/\omega\} \{1 - \cos \omega(t-2)\} \text{ m}$$

$$z_I(t) = 20000 \text{ m}$$

The trajectory 2 equation can be modified so that instead of the target performing a pull-up maneuver in the inertial y-direction the target motion is in the negative inertial z-direction. In this case, the target turns in toward the FLIR image plane, and three distinct ellipsoidal intensity patterns are projected onto the FLIR plane. With the x equations being as previously described, the motion of the target is defined by:

$$\dot{y}_I(t) = 0.0 \text{ m/sec}$$

$$\dot{z}_I(t) = -1000.0 \sin\{\omega(t-2)\} \text{ m/sec}$$

The inertial position is given by:

$$y_I(t) = 500.0 \text{ m}$$

$$z_I(t) = z_I(t_0) - \{1000/\omega\}\{1 - \cos \omega(t-2)\} \text{ m}$$

ω is set at various constant values in different simulations, to evaluate the performance of the tracker against targets exhibiting varying degrees of maneuverability. In this research, the performance of the tracker against targets performing 2 and 5 g turns was evaluated. ($\omega = 0.0196$ and 0.049 radians/sec respectively). Note that this trajectory represents a step change to a constant g maneuver. Although realistically such changes do not occur, for ease of implementation this method was used. However, this also implies that the filter is being required to track a harsher maneuver than the more realistic pull-up maneuver, in which ω builds up smoothly from zero to a constant rate, and thus better performance should be achieved in the more realistic environment.

Trajectory 3 was motivated by the desire to evaluate not only how the filter responds when the target being tracked initiates a maneuver, but also how the filter responds when the target terminates the maneuver.

Trajectory 3 is initially the same as trajectory 2, with a two-g pull-up maneuver being initiated at $t = 2.0$ sec. With this maneuver the target has an inertial velocity at $t = 3.5$ sec of

$$\dot{x}_I(t = 3.5) = -999.58 \text{ m/sec}$$

$$\dot{y}_I(t = 3.5) = 29.069 \text{ m/sec}$$

$$\dot{z}_I(t = 3.5) = 0. \text{ m/sec}$$

The target terminates the pull-up maneuver at $t = 3.5$ sec and continues at this constant velocity for the remainder of the simulation.

The final trajectory, trajectory 4, was designed as the most realistic trajectory, and also to evaluate the performance of the filter when the dynamic shape of the target intensity pattern is changing. Trajectory 4 is initialized identically to trajectory 2. However, at $t = 2$ sec, a two-g turn is initiated, with the target turning toward the FLIR plane displaying motion in all inertial directions (see Figure 8).

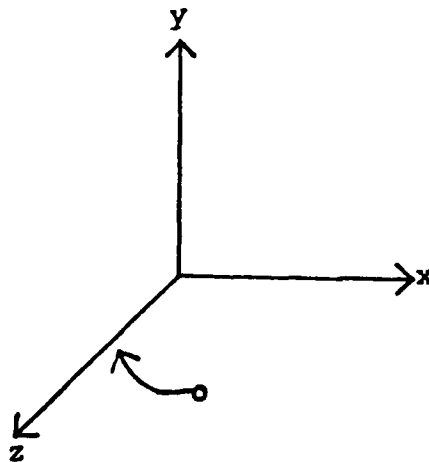


Figure 8. Trajectory 4.

This maneuver can be derived by the two coordinate system transformations shown in Figure 9.

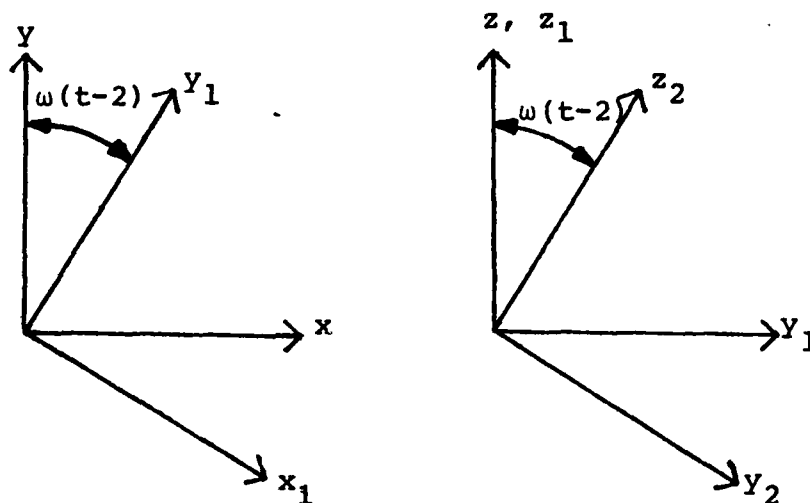


Figure 9. Out-of-Plane Coordinate Frame Rotations

Explicitly writing out the coordinate transformations yields:

$$\begin{bmatrix} \dot{x}_I(t) \\ \dot{y}_I(t) \\ \dot{z}_I(t) \end{bmatrix} = \begin{bmatrix} 1 & 0 & 0 \\ 0 & A & B \\ 0 & -B & A \end{bmatrix} \begin{bmatrix} A & B & 0 \\ -B & A & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} V \\ 0 \\ 0 \end{bmatrix}$$

where

$$A = \cos \omega (t-2)$$

$$B = \sin \omega (t-2)$$

Performing the matrix multiplication gives the velocity equations as:

$$\dot{x}_I(t) = -v\{\cos \omega(t-2)\} \text{ m/sec}$$

$$\dot{y}_I(t) = v\{\cos \omega(t-2)\} \sin \omega(t-2) \text{ m/sec}$$

$$\dot{z}_I(t) = -v\{\sin^2 \omega(t-2)\} \text{ m/sec}$$

and the inertial target position is:

$$x_I(t) = x_I(t_0) - 2000 - \{(1000/\omega) (\sin \omega(t-2))\}$$

$$y_I(t) = y_I(t_0) + \{(\frac{1000}{2\omega}) (\sin^2 \omega(t-2))\}$$

$$z_I(t) = z_I(t_0) - 1000 \{(\frac{t-2}{2}) - \frac{1}{4\omega} \sin(2\omega(t-2))\}$$

where in this simulation $v = -1000$ m/sec. The out-of-plane angle associated with this maneuver can be found by

$$\text{op angle} = \tan^{-1} \frac{z_I(t)}{y_I(t)}$$

2.4 Measurement Model

With the motion of the target defined, the next step is to define the intensity function generated by the target and project that function onto the FLIR image plane. For distant targets, the intensity pattern projection onto the FLIR image plane is well approximated by a bivariate Gaussian function with circular equal intensity contours (5:223). However, for closer range targets, Harnly and Jensen, ref 6, found an elliptically shaped pattern was a better representation of the true intensity pattern. This intensity function is

$$I(x,y) = I_{\max} \exp \left[-0.5 \{ (x-x_{\text{peak}}) (y-y_{\text{peak}}) \} \{ \underline{P} \}^{-1} \begin{Bmatrix} x-x_{\text{peak}} \\ y-y_{\text{peak}} \end{Bmatrix} \right] \quad (2-9)$$

where the variables, as shown in Figure 10, are

I_{\max} = maximum target intensity

$x_{\text{peak}}, y_{\text{peak}}$ = coordinates of peak intensity function

σ_v, σ_{pv} = eigenvalues of \underline{P} ; corresponding to the ellipse semimajor axis along the velocity vector and the semiminor axis perpendicular to the velocity vector respectively.

$v_{\perp \text{LOS}}$ = velocity component of the target perpendicular to the line of sight from the FLIR image plane to the target

θ = orientation angle of $v_{\perp \text{LOS}}$ in the FLIR plane, aligned with the semimajor axis of the intensity pattern

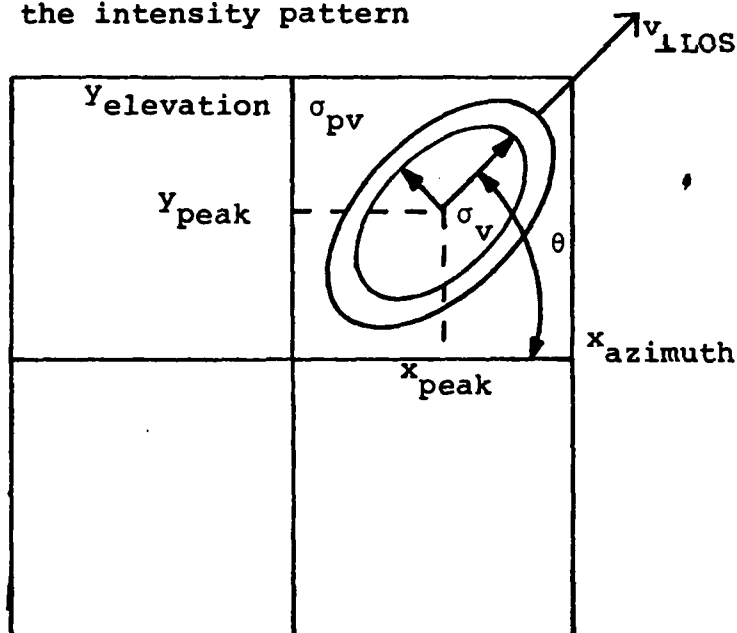


Figure 10. Image Intensity Characteristics

For this research, a pixel dimension of 20μ rads is used. The movement of the point of maximum intensity on the FLIR image plane defined by x_{peak} and y_{peak} is determined by $\alpha(t)$ and $\beta(t)$, expressed in μ rads, respectively after the effects of tracker controller action have been accounted for. Therefore, the 8×8 FLIR FOV will be 160μ rads wide in azimuth and 160μ rads wide in elevation. Because of this small FOV, angular displacement of the target from the FLIR FOV can be approximated by linear displacement on the FLIR image plane. Similarly, angular velocity closely approximates linear velocity in the FLIR image plane (6:24). Thus, angular measurements were used.

The intensity pattern on the FLIR image plane is produced in several steps. From the simulation of the inertial position and velocity of the missile, the azimuth velocity (x velocity in Figure 11), elevation velocity (y velocity in Figure 11), and speed (the magnitude of the velocity vector in inertial space), were computed in rad/sec. The azimuth and elevation velocity then define the missile velocity component perpendicular to the line of sight, $v_{\perp LOS}$, and the ratio of $v_{\perp LOS}$ and speed is the cosine of the out of place angle, θ . Figure 11 shows the geometry involved (6:24). Explicitly, the relationships are:

$$\cos \theta = \frac{\dot{\alpha}(t)}{|v_{\perp LOS}|}$$

$$\sin \theta = \frac{\dot{\beta}(t)}{|v_{\perp LOS}|}$$

where

$$|v_{\perp LOS}| = \{\dot{\alpha}^2(t) + \dot{\beta}^2(t)\}^{1/2}$$

and

$$\cos \gamma = \frac{|v_{LOS}|}{|v|}$$

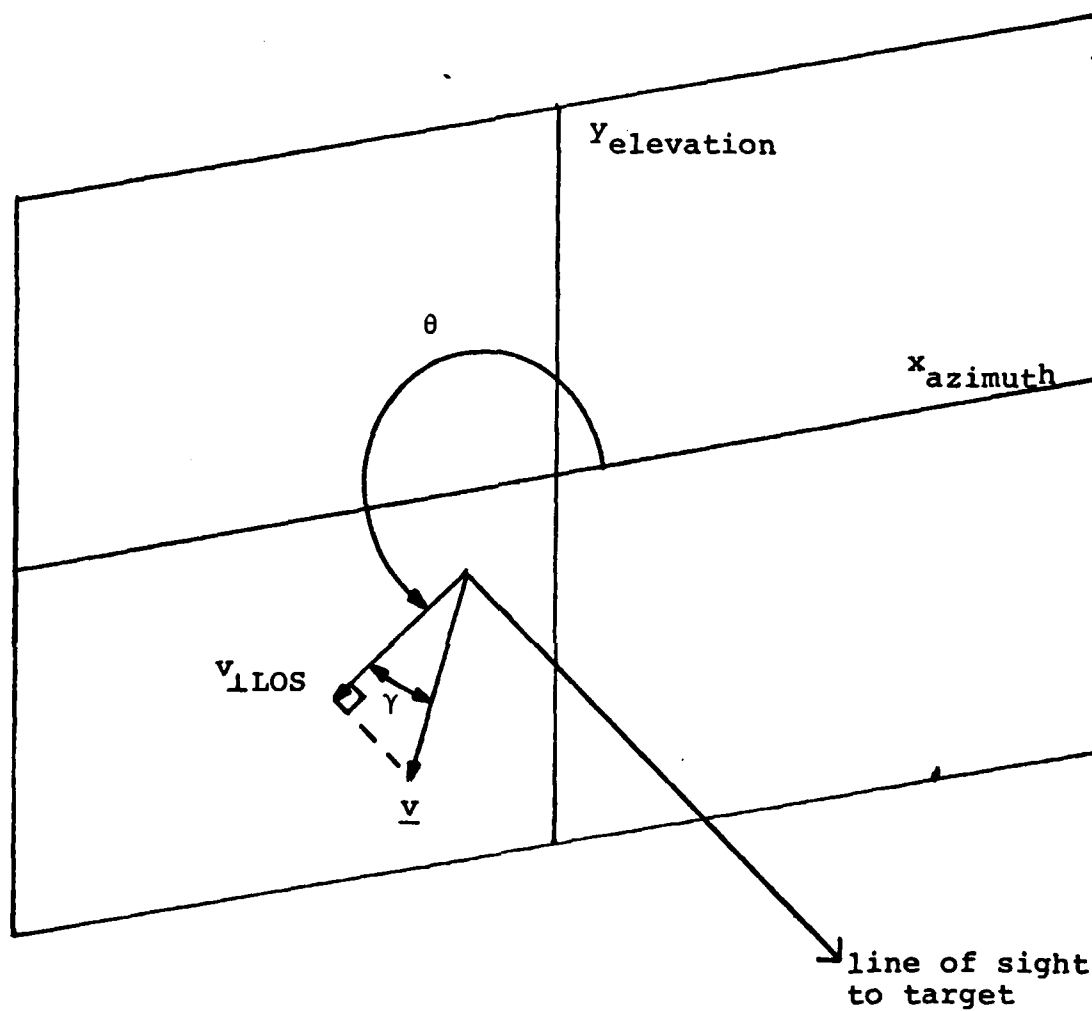


Figure 11. Image Projection

With $\cos \gamma$ determined, the semimajor axis of the missile can be projected onto the FLIR image plane. Referring to Figure 12, where δ is the length of the

semimajor axis in meters, the length of the semimajor axis parallel to the FLIR plane is:

$$\sigma'_p = \delta \cos \gamma \text{ m}$$

By approximating linear distance on the FLIR plane as the angular displacement, this distance can be expressed as

$$\sigma_p = \psi / .00002 \text{ pixels} \quad (2-10)$$

where

$$\psi = \frac{\sigma'_p}{r}$$

.00002 = conversion factor from radians to pixels

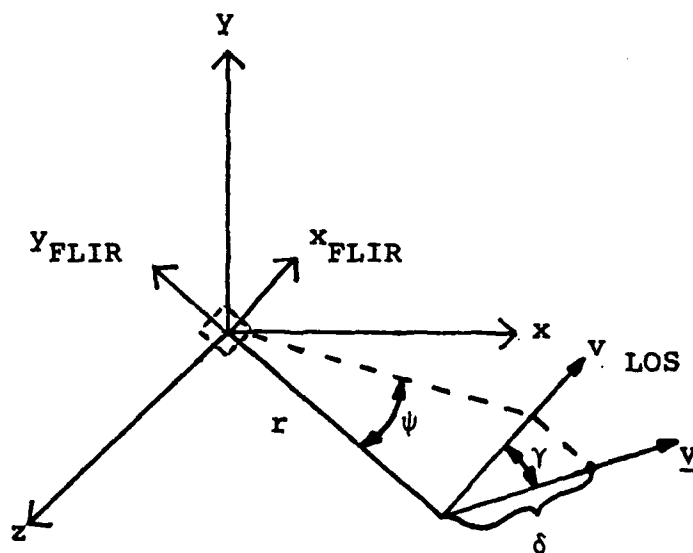


Figure 12. Semimajor Axis Projection

To make use of information already computed in the

simulation and with $\cos \gamma = 1$ at $t = 0$ for the trajectories used, Equation (2-10) becomes

$$\sigma_v = \frac{(\cos \gamma)(\sigma_{vI})(r_I)}{r} \text{ pixels} \quad (2-11)$$

where

r_I = initial target range

σ_{vI} = initial length of semimajor axis in pixels

The radius of the circular missile IR cross-section is retained as the semiminor axis and similar to the development of Equation (2-11) the distance on the FLIR image plane is given by

$$\sigma_{pv} = \frac{(\sigma_{pvI})(r_I)}{r} \text{ pixels} \quad (2-12)$$

where

σ_{pvI} = initial length of semiminor axis in pixels

With these parameters established, the intensity at any point in the image plane can be computed. This calculation is performed in the image ellipse coordinate system. The intensity function is then (6:27)

$$I(x,y) = I_{\max} \exp \left\{ -0.5 [\Delta x \Delta y] \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_{pv}^2 \end{bmatrix}^{-1} \begin{bmatrix} \Delta x \\ \Delta y \end{bmatrix} \right\} \quad (2-13)$$

where

$$\Delta x = (x - x_{\text{peak}}) \cos \theta + (y - y_{\text{peak}}) \sin \theta$$

$$\Delta y = (y - y_{\text{peak}}) \cos \theta - (x - x_{\text{peak}}) \sin \theta$$

θ = rotation angle between FLIR axis and image ellipse coordinate axis

The average intensity for any pixel, as measured by the FLIR, is the integral of the apparent target intensity function over the pixel area divided by the area of the pixel, corrupted by FLIR and background noise. To approximate this integral, the intensity function was averaged over twenty five equally spaced points within each pixel. To complete the simulation, noise corresponding to FLIR and background noise was added to each pixel. For one of the 64 pixels in the l-th row and m-th column the average intensity as measured by the FLIR is represented as:

$$z_{lm}(t_i) = \frac{1}{25} \sum_{k=1}^5 \sum_{j=1}^5 I_{\max} \exp \left\{ -0.5 \begin{bmatrix} \Delta x_{lk} & \Delta x_{mj} \end{bmatrix} \begin{bmatrix} \sigma_v^2 & 0 \\ 0 & \sigma_{vp}^2 \end{bmatrix}^{-1} \begin{bmatrix} \Delta x_{lk} \\ \Delta y_{mj} \end{bmatrix} \right\} + n_{lm}(t_i) \quad (2-14)$$

The noise term, $n_{lm}(t_i)$, is based on the research of Harnly and Jensen, ref 6, who documented the existence of spatial correlations of background noise in each data frame with nonnegligible spatial correlations between each pixel and its closest two neighboring pixels in each direction (6:19). The 64 measurements are first arranged into a vector. Then, the covariance matrix, "R", for the zero-mean white Gaussian noise, n , consisting of components seen in (2-14), is of dimension 64x64. With the spatial correlation matrix, R, known, realizations of the noise vector can be produced by performing a Cholesky square root decomposition of R and post-multiplying it by a vector of independent, white Gaussian noises, each of zero mean and variance

of one. The noise vector is

$$\underline{v}(t_i) = \sqrt{R} \underline{v}'(t_i) \quad (2-15)$$

where

$\underline{v}'(t_i)$ = white Gaussian noise vector with independent scalar noises and statistics

$$E\{\underline{v}'(t_i)\} = 0$$

$$E\{\underline{v}'(t_i) \underline{v}'^T(t_j)\} = \underline{I} \delta_{ij}$$

The model given in Equation (2-15) is used because the noise \underline{v}' is readily simulated via repeated independent calls to a Gaussian random number generator. Thus, the covariance of $\underline{v}(t_i)$ is equal to the correlation matrix.

$$E\{\underline{v}(t_i) \underline{v}^T(t_j)\} = E\{\sqrt{R} \underline{v}'(t_i) \underline{v}'^T(t_j) \sqrt{R}^T\} = R \delta_{ij}$$

As justified in the analysis by Harnly and Jensen, temporal correlation of the background and FLIR noises are assumed negligible. The noise array vector is then added to the input array, as in Equation (2-14), to create the measurement array for the correlator/Kalman filter.

Equation (2-14) represents the measurement array for a single hot spot target. For multiple hot spot targets, three Gaussian hot spots with elliptical constant intensity contours and parallel semimajor axes were used in this study. The measurement is then the summation of three double summation terms of the form given in (2-14) instead of one, plus the noise as in (2-14).

2.5 Projection of Multiple Hot Spots

The location of the aircraft center of mass (COM)

in FLIR coordinates can be determined, as previously described, and the method for projecting the intensity pattern onto the FLIR plane is the same for multiple and single hot spot cases because the assumption is made that the semimajor axes of the ellipsoids are parallel. Thus, by calculating the angular orientation of one of the hot spots the orientation of all of the hot spots is known. Therefore, the only remaining information required for the projection is the coordinates of the center of each ellipsoid in the FLIR image plane. This requires that the distance from the COM of the aircraft to the center of each ellipsoid be known in inertial frame coordinates and transformed into FLIR frame coordinates. For single hot spot targets, the center of the intensity ellipsoid is assumed to coincide with the aircraft COM and no additional calculations are required.

With the orientation of the FLIR image plane relative to an inertial frame being as shown in Figure 13, and the angles α and β known, as calculated in the trajectory model, unit vectors in the directions referred to as \vec{e}_β and $\vec{e}_{z\alpha}$ can be determined. Notice that in FLIR frame coordinates $\vec{e}_\beta = \vec{e}_{y \text{ FLIR}}$ and $\vec{e}_{z\alpha} = -\vec{e}_{x \text{ FLIR}}$. This coordinate frame was selected instead of using $\vec{e}_{x \text{ FLIR}}$ directly for ease in implementing the required coordinate frame transformations (Appendix A).

The \vec{e}_β - $\vec{e}_{z\alpha}$ plane is then translated in inertial space so that the origin of this plane coincides with the aircraft COM (Figure 14).

From the trajectory model, the velocity vector of the aircraft is known. Since the location of the hot spots in a body fixed frame can be readily determined, the velocity vector, which by assumption will always point out the nose of the aircraft, can be used to define

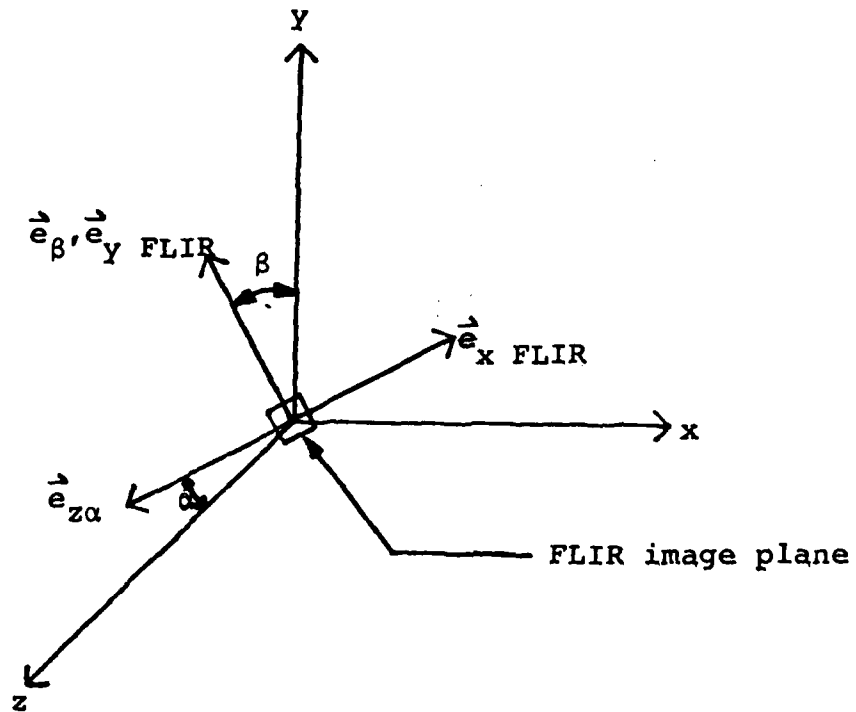


Figure 13. $\vec{e}_\beta - \vec{e}_{z\alpha}$ Plane

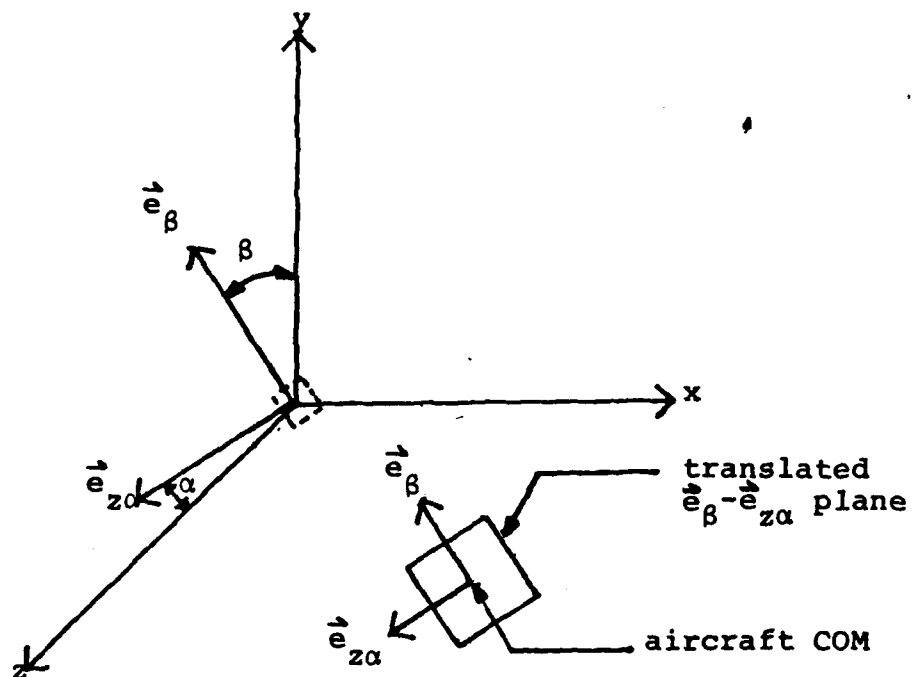


Figure 14. $\vec{e}_\beta - \vec{e}_{z\alpha}$ Plane Translation

one axis of a coordinate frame with its origin lying at the aircraft COM, referred to as the H-frame. Thus, \vec{e}_{Hx} is defined to be a unit vector in the direction of the velocity vector and forms one axis of the H-frame. Performing a cross-product of the velocity vector with a unit vector in the inertial y direction, \vec{j} , will produce a vector normal to \vec{j} and \underline{v} , which is normalized to form a unit in the direction of the second axis of the H-frame, \vec{e}_{Hy} . Note that this axis will always lie in the horizontal plane. The third axis of the H-frame is subsequently calculated by crossing the two H-frame axis vectors, to produce a vector normal to the \vec{e}_{Hx} - \vec{e}_{Hy} plane, which when normalized becomes \vec{e}_{Hz} . The unit vectors of the translated \vec{e}_{β} - $\vec{e}_{z\alpha}$ frame and the H-frame with origins located at the aircraft COM, are shown in Figure 15.

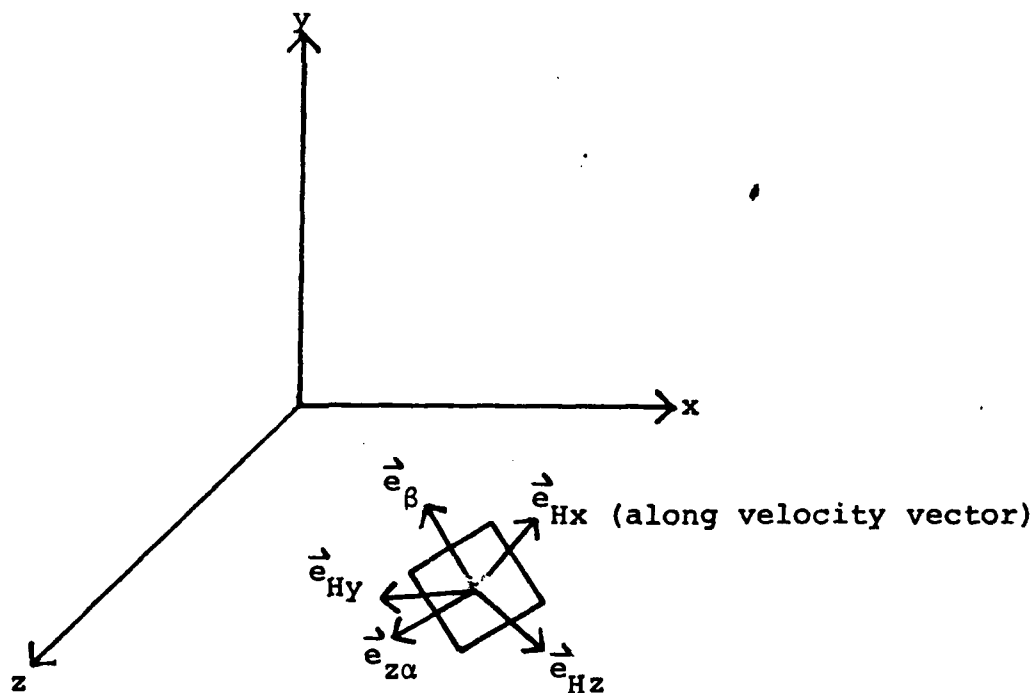


Figure 15. \vec{e}_{β} - $\vec{e}_{z\alpha}$ and H-Frame Unit Vectors

Let the body frame, B, have the x-axis out the aircraft nose, the y-axis out the aircraft right wing, and the z-axis out the aircraft belly, and assume that initially the aircraft is oriented such that the aircraft body frame lies with the plane of the aircraft wings in the $\vec{e}_{Hx}-\vec{e}_{Hy}$ plane. Specifically for the three hot spot target model used, the initial location of the intensity functions are shown in Figure 16.

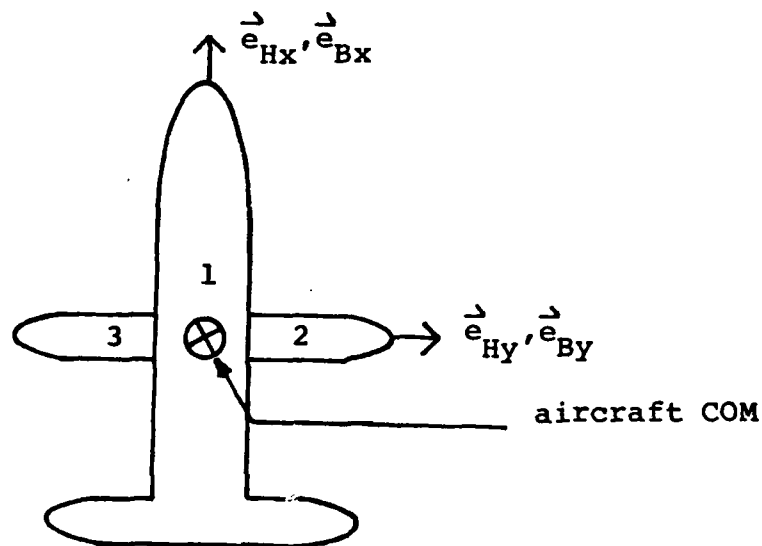


Figure 16. Initial Ellipsoidal Centers

where

$\vec{e}_{Hz}, \vec{e}_{\beta z}$ into the page

①, ②, and ③, location of the center of each intensity ellipsoid, in relation to the aircraft COM.

By assumption, the center of ellipsoid 1 will

lie along the \vec{e}_{Hx} axis. The ellipsoidal centers along the \vec{e}_{Hy} axis will remain along that axis unless the aircraft performs a roll maneuver, in which case their location can be determined as shown in Figure 17. In order to simulate a roll maneuver in this thesis, a constant roll-rate, ω , was used. Using a constant roll-rate implies a step change from a wing-level trajectory to a constant roll-rate rather than a smooth build up to the desired rate. This method was used for ease in implementing the simulation. However, this also requires the filter to track a maneuver which is harsher than realistically would be encountered and thus the results obtained are for a worst case scenario.

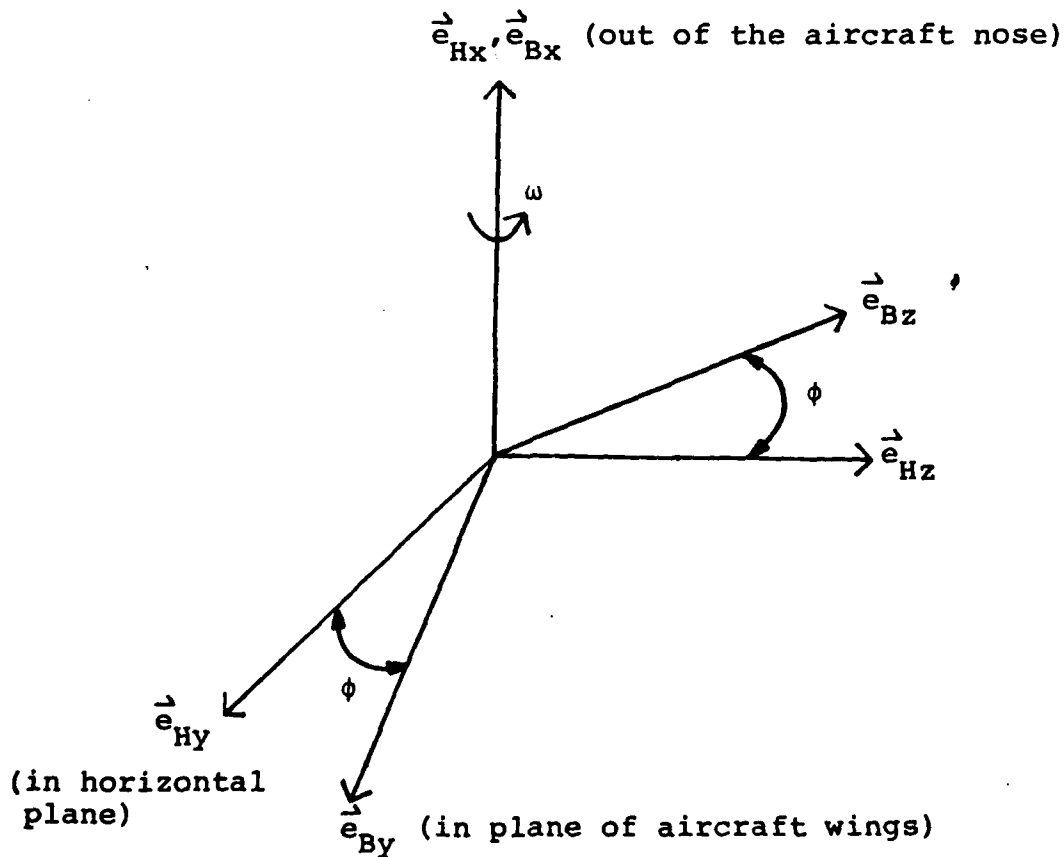


Figure 17. Roll Maneuver Geometry

Since the intensity function centroids are offset in the $\vec{e}_{Bx}-\vec{e}_{By}$ plane, the only calculation required to determine the direction of the ellipsoidal centers is

$$\vec{e}_{By} = \cos \phi \vec{e}_{Hy} + \sin \phi \vec{e}_{Hz} \quad (2-16)$$

where

ϕ is the roll angle measured from \vec{e}_{Hy} to \vec{e}_{By} . In simulations for this thesis, ϕ is the result of a constant roll rate ω , $\phi = \omega t$

Thus, the location of the three hot spots relative to the H-frame can always be determined.

The final step is to convert this distance to the equivalent offset distance on the FLIR image plane. Let δ represent the distance in meters from the aircraft COM to the center of ellipsoid 2, lying in the positive e_{By} direction. This is depicted in Figure 18.

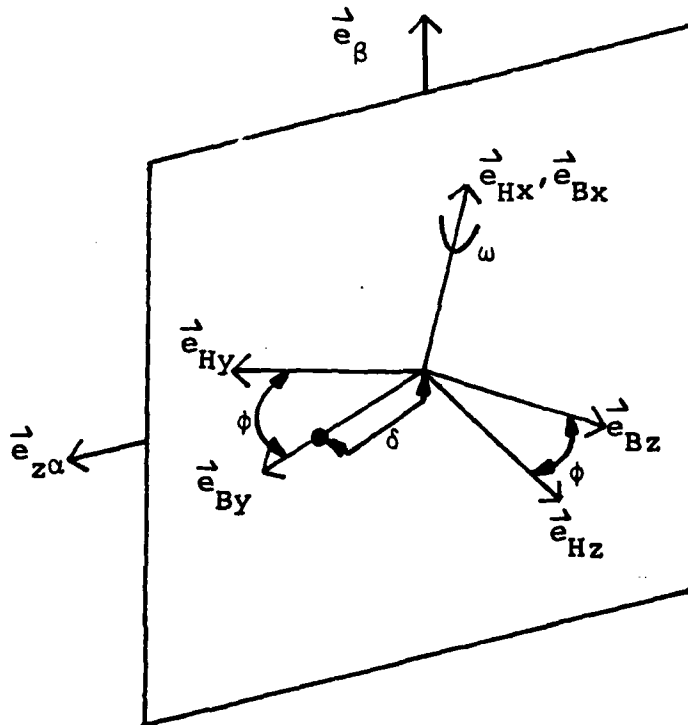


Figure 18. Hot Spot Offset

By taking the following dot products, $\delta\{\vec{e}_{By} \cdot \vec{e}_{z\alpha}\}$ and $\delta\{\vec{e}_{By} \cdot \vec{e}_{\beta}\}$, the projection of the ellipsoidal displacement onto the translated $\vec{e}_{\beta} - \vec{e}_{z\alpha}$ plane is determined, as shown in Figure 19.

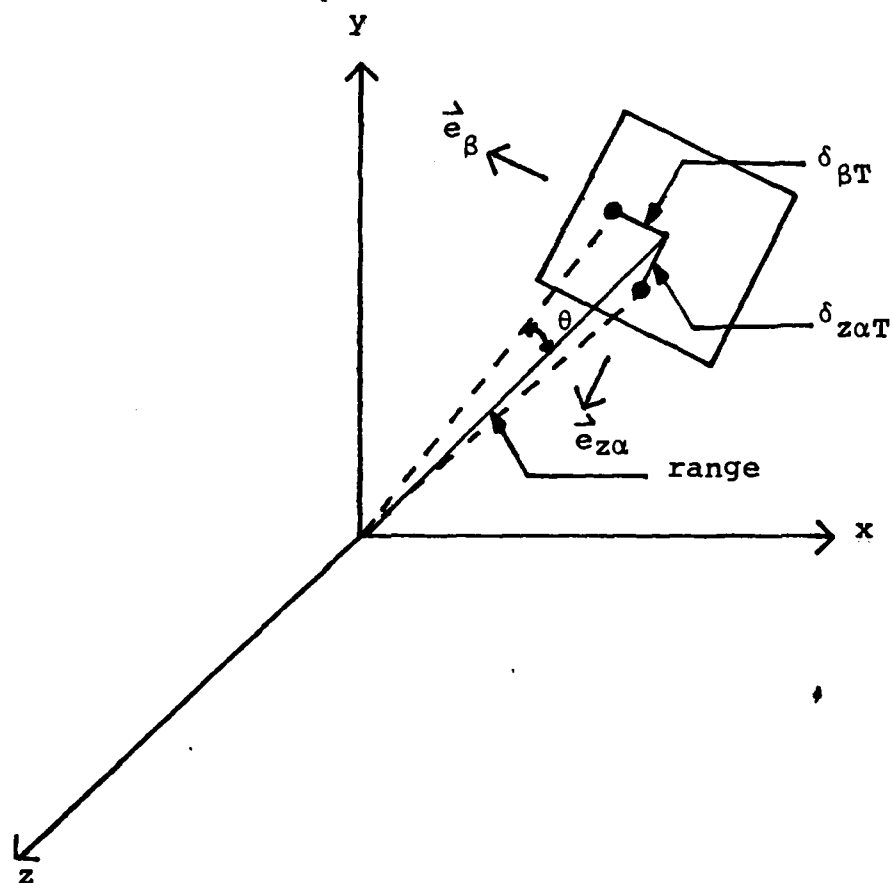


Figure 19. Distance Translated in $\vec{e}_{\beta} - \vec{e}_{z\alpha}$ Plane

where

$$\delta_{\beta T} = \delta(\vec{e}_{By} \cdot \vec{e}_{\beta}) = \text{distance along translated } \vec{e}_{\beta} \text{ axis}$$

in meters.

$$\delta_{z\alpha T} = \delta(\vec{e}_{By} \cdot \vec{e}_z) = \text{distance along translated} \\ \vec{e}_{z\alpha} \text{ axis in meters}$$

With $\delta_{\beta T}$ and $\delta_{z\alpha T}$ known, this distance is converted into distance on the FLIR image plane. Since the translated \vec{e}_{β} - $\vec{e}_{z\alpha}$ plane is normal to the range vector, the angular displacement of the hot spot from the center of the FLIR FOV can be used to approximate the linear displacement in the FLIR plane. In the \vec{e}_{β} direction,

$$\delta_{\beta} = \tan \theta \approx \theta = \frac{\delta_{\beta T}}{r} \quad (2-17)$$

where

θ = displacement angle as shown in Figure 19.

r = range

Equation (2-17) gives the distance, δ_{β} , in radians, which is converted to pixels by dividing by .00002 radians/pixel. Displacement in the \vec{e}_{β} direction is added in the y_{FLIR} direction to the coordinates of the aircraft COM to determine the y_{FLIR} location of the hot spot. Displacement in the $\vec{e}_{z\alpha}$ direction is likewise calculated and subtracted from the coordinates of the aircraft COM to determine the x_{FLIR} location of the hot spot because of the axis orientation shown in Figure 13. Thus, the coordinates of the ellipsoidal centers in the FLIR image plane are known. To demonstrate this projection, a sample trajectory was run with initial coordinates

$$x_I = y_I = z_I = 10,000 \text{ m}$$

and constant velocity

$$\dot{x}_I(t) = \dot{y}_I(t) = \dot{z}_I(t) = -500.0 \text{ m/sec} \quad (2-19)$$

The target is moving directly at the FLIR image plane therefore,

$$\dot{\alpha}(t) = \dot{\beta}(t) = 0$$

for the entire trajectory. Hot spots were initially positioned 1 meter forward of the aircraft COM in the \vec{e}_{Hx} direction and (\pm) .5 meters in the \vec{e}_{Hy} direction. Two simulations were used to demonstrate this projection model. The first run, see Figure 20, shows that for a roll of π radians, hot spot 1 stays centered in the $\vec{e}_{\beta}-\vec{e}_{z\alpha}$ plane, while hot spots 2 and 3 roll through π radians and remain positioned π radians apart on the $\vec{e}_{\beta}-\vec{e}_{z\alpha}$ plane. As the aircraft approaches the tracker location, the distance from the aircraft COM to the hot spots as projected onto the FLIR plane will increase which accounts for the offset semicircles seen in Figure 20. The second run also initialized as in (2-18) with velocity components as in (2-19), was used to show the proper displacement on the $\vec{e}_{\beta}-\vec{e}_{z\alpha}$ plane. During this run, shown in Figure 21, the aircraft conducted a 2π roll about its velocity vector with the hot spots positioned as in the first run. Substituting the initial conditions from Equation (2-18) into Equation (2-17) yields the initial offset distance for hot spots 2 and 3 from the aircraft COM.

$$\delta_{z\alpha} = \left[\frac{(\pm)(.5)}{\{(3)(10000^2)\}^{1/2}} \right] \left[\frac{1}{.00002} \right] = (\pm) 1.44 \text{ pixels}$$

For a 5 second simulation Equation (2-19) can also be

used to determine the inertial position of the aircraft at $t = 5.0$. Thus, Equation (2-17) can again be used to determine final offsets of the hot spots.

$$\delta_{z\alpha} = \left[\frac{(\pm)(.5)}{\{(3)(7500^2)\}^{1/2}} \right] \left[\frac{1}{.00002} \right] = (\pm) 1.93 \text{ pixels}$$

As shown in Figure 21, hot spots 2 and 3 roll through 2π radians with initial and final offset distance as calculated and hot spot 1 remains centered in the plane. For a detailed description of the projection model equations, see Appendix A.

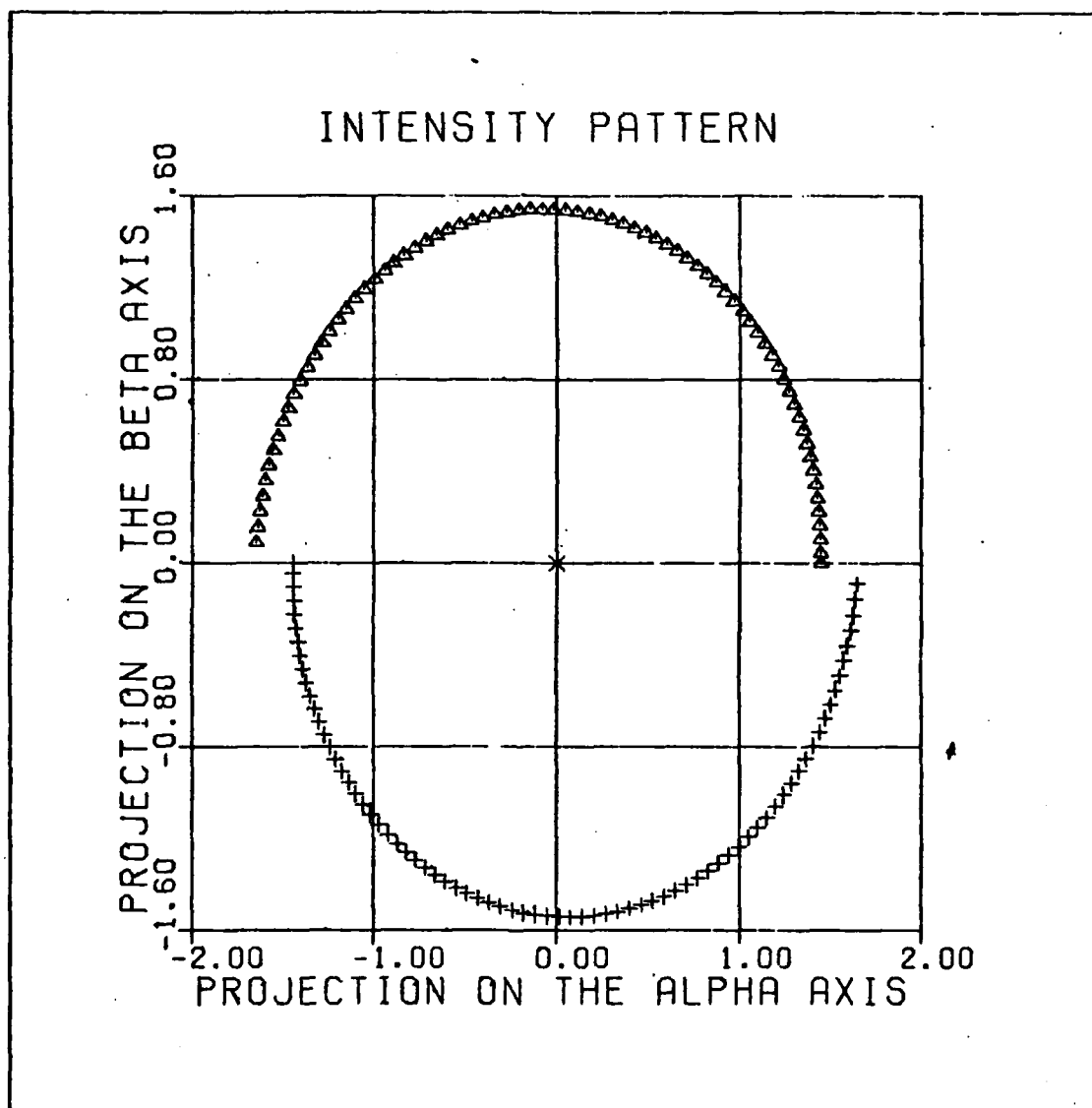


Figure 20. π Radians Roll

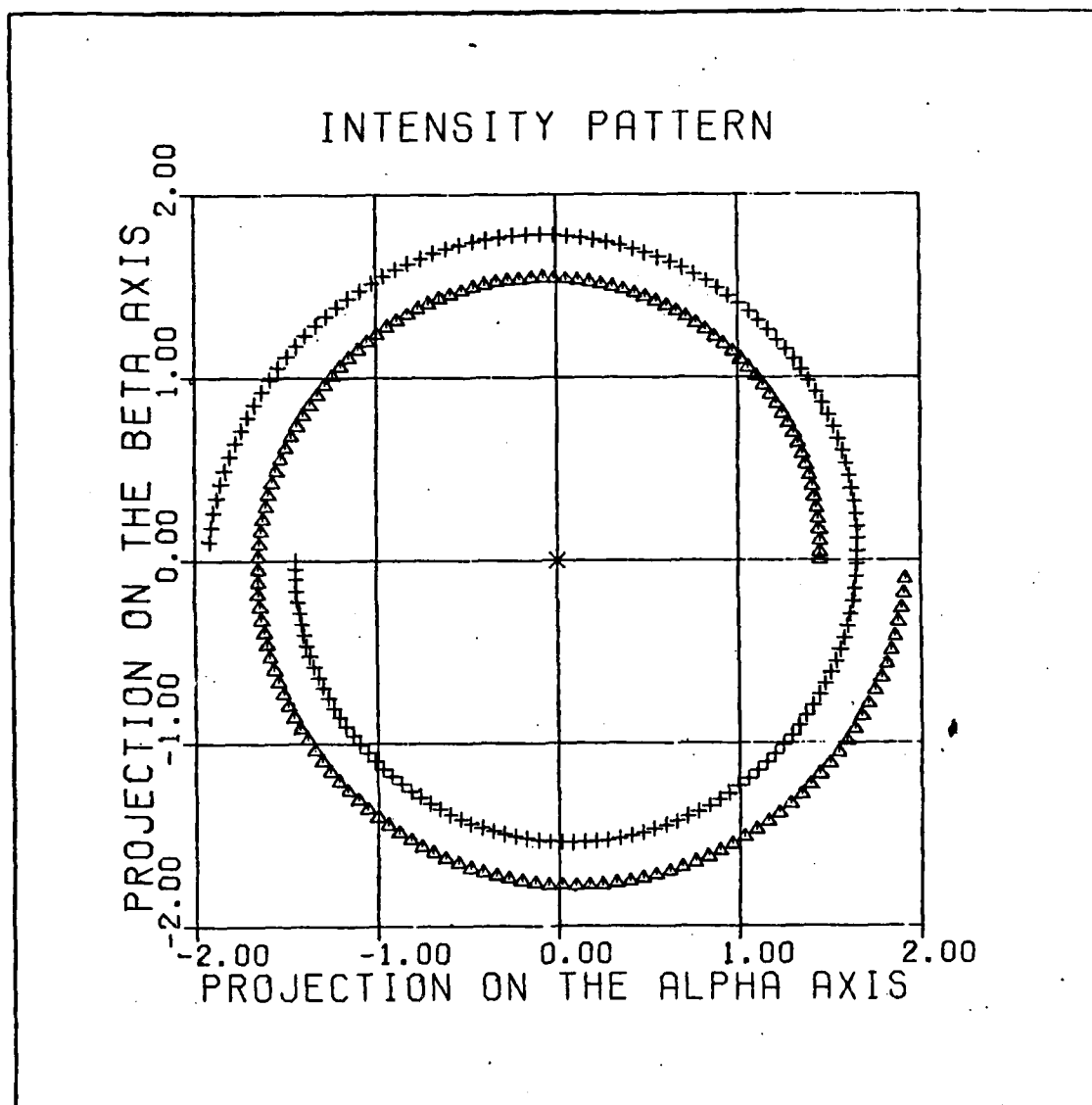


Figure 21. 2π Radians Roll

III. Kalman Filter

3.1 Introduction

The primary focus of Rogers' thesis (8) was to develop an algorithm capable of indentifying, in real-time, the intensity function of the target. As such, the ability of the Kalman filter to track a realistic, highly dynamic target was not of primary concern, so a four-state Kalman filter, consisting of estimates for the x and y position due to true target dynamics and the x and y position due to atmospheric, which sum together to form the apparent target centroid location as,

$$\begin{matrix} x \\ \text{centroid} \end{matrix} = \begin{matrix} x \\ \text{dynamics} \end{matrix} + \begin{matrix} x \\ \text{atmospherics} \end{matrix} \quad (3-1)$$

and similarly in the y-direction, was used. Such a model is acceptable if the target is not highly dynamic and does not leave the FLIR FOV in one sample period. However, in practical applications a missile can travel many times the width of the FLIR FOV in one sample period. For example, a typical missile at a range of 10 km can travel approximately 80 pixels per sample period and a FLIR with an 8x8 pixel FOV and no predictive capability will not be able to keep the target within its FOV, resulting in loss of target track (6:69). Therefore, the tracker must have a predicted target position for the next sample period.

Although a six-state filter which additionally estimates target velocity in both directions might be appropriate in certain situations, previous research efforts have shown that in order to achieve accurate target position information acceleration estimates in both directions must be incorporated into the filter model as well, resulting in an eight-state filter (6).

Of the acceleration models most commonly used, first order Gauss-Markov and constant turn-rate models have yielded the best results, with the constant turn-rate model proving more representative of the tracking scenarios being investigated (7,16). However, the performance enhancement achieved by using the non-linear constant turn-rate model is not so significant as to justify the increased computational burden associated with its implementation over the linear Gauss-Markov model (11). Therefore, a first-order Gauss-Markov acceleration model was selected for this study. As in Rogers' four-state filter, this filter uses the Gauss-Markov model developed by Mercier for the atmospheric jitter process (10).

3.2 Acceleration and Atmospheric Models

Target acceleration and atmospheric jitter are modeled as stationary, first-order Gauss-Markov processes, generated as the outputs of first-order lags driven by white Gaussian noises. The differential equation describing such a process is

$$\dot{x}(t) = -\frac{1}{T}x(t) + w(t) \quad (3-2)$$

and

$$E\{w(t)\} = 0$$

$$E\{w(t)w(t+\tau)\} = \frac{2\sigma^2}{T} \delta(\tau)$$

where

$x(t)$ = the Gauss-Markov process with initial conditions determined from a priori knowledge of the process

T = correlation time of $x(t)$

$w(t)$ = white Gaussian noise process

σ = root mean square value of $x(t)$

Throughout the development of this chapter, the subscripts a, d, and f, will be used to represent atmospherics, dynamics, and filter, respectively. For the atmospheric position, the correlation time constant, T_{af} , as developed by Mercier (10) will be used. The acceleration time constant, T_{df} , is a parameter to be established during the off-line filter tuning process although on-time estimation techniques could be employed (13:Chapter 10). Equation (3-2) serves as the stochastic model upon which the filter is based.

3.3 State Space Model

The eight-state filter will now be put into state space notation. The dynamics model in each direction is

$$\begin{aligned}\dot{x}_{df}(t) &= v_{df}(t) \\ \dot{v}_{df}(t) &= a_{df}(t) \\ \dot{a}_{df}(t) &= -\frac{1}{T_{df}} a_{df}(t) + w_{df}(t)\end{aligned}\tag{3-3}$$

and

$$\begin{aligned}E\{w_{df}(t)\} &= 0 \\ E\{w_{df}(t)w_{df}(t+\tau)\} &= \frac{2\sigma_{df}^2}{T_{df}} \delta(\tau)\end{aligned}$$

where

T_{df} = correlation time assumed for target acceleration

σ_{df}^2 = assumed target acceleration process variance

The atmospheric disturbance model in each direction is

$$\dot{x}_{af}(t) = -\frac{1}{T_{af}} x_{af}(t) + w_{af}(t) \quad (3-4)$$

and

$$E\{w_{af}(t)\} = 0$$

$$E\{w_{af}(t)w_{af}(t+\tau)\} = \frac{2\sigma_{af}^2\delta(\tau)}{T_{af}}$$

where

T_{af} = correlation time assumed for atmospheric jitter

σ_{af}^2 = assumed atmospheric jitter process variance

Augmenting the atmospheric states to the dynamic states, the propagation equation upon which the Kalman filter is based is formed:

$$\dot{\underline{x}}_f(t) = \underline{F}_f(t)\underline{x}_f(t) + \underline{G}_f(t)\underline{w}_f(t) \quad (3-5)$$

where

$\underline{F}_f(t)$ = time-invariant filter plant matrix

$\underline{x}_f(t)$ = filter state vector

x_1 = azimuth position

x_2 = elevation position

x_3 = azimuth velocity = \dot{x}_1

x_4 = elevation velocity = \dot{x}_2

x_5 = azimuth acceleration = \dot{x}_3

x_6 = elevation acceleration = \dot{x}_4

x_7 = azimuth atmospheric disturbance

x_8 = elevation atmospheric disturbance

\underline{G}_f = constant 8x4 noise distribution matrix

$\underline{w}_f(t)$ = 4-dimensional white Gaussian noise composed
of $w_1 = w_{dfx}$ and $w_2 = w_{dfy}$ as in (3-3) and
 $w_3 = w_{afx}$ and $w_4 = w_{afy}$ as in (3-4).

Substituting Equations (3-3) and (3-4) into Equation
(3-5) yields:

$$\dot{\underline{x}}_f(t) = \begin{bmatrix} 0 & 0 & 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 1 & 0 & 0 \\ 0 & 0 & 0 & 0 & -\frac{1}{T_{df}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{df}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{af}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -\frac{1}{T_{af}} \end{bmatrix} \underline{x}_f(t)$$

$$+ \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} w_1(t) \\ w_2(t) \\ w_3(t) \\ w_4(t) \end{bmatrix}$$

(3-6)

where $w_1(t)$ through $w_4(t)$ are zero-mean, independent white Gaussian noises with

$$E\{\underline{w}_f(t)\underline{w}_f^T(t+\tau)\} = \underline{Q}_f\delta(\tau)$$

and

$$\underline{Q}_f = \begin{bmatrix} \frac{2\sigma_{df}^2}{T_{df}} & 0 & 0 & 0 \\ 0 & \frac{2\sigma_{df}^2}{T_{df}} & 0 & 0 \\ 0 & 0 & \frac{2\sigma_{af}^2}{T_{af}} & 0 \\ 0 & 0 & 0 & \frac{2\sigma_{af}^2}{T_{af}} \end{bmatrix} \quad (3-7)$$

3.4 State Propagation

The Kalman filter must propagate its state estimate vector and conditional covariance matrix from one sample time to the next. The standard propagation equations for the case of zero control inputs over a sample interval are:

$$\hat{x}_f(t_{i+1}^-) = \phi_f(t_{i+1}, t_i) \hat{x}_f(t_i^+) \quad (3-8)$$

$$\underline{P}_f(t_{i+1}^-) = \phi_f(t_{i+1}, t_i) \underline{P}_f(t_i^+) \phi_f^T(t_{i+1}, t_i) + \quad (3-9)$$

$$+ \int_{t_i}^{t_{i+1}} \phi_f(t_{i+1}, \tau) \underline{G}_f(\tau) \underline{Q}_f(\tau) \underline{G}_f^T(\tau) \phi_f^T(t_{i+1}, \tau) d\tau$$

where

$\phi_f(t_{i+1}, t_i)$ = filter state transition matrix

$\underline{P}_f(t_i^+)$ = conditional state covariance matrix
from measurement update at time t_i

$\underline{P}_f(t_{i+1}^-)$ = conditional state covariance matrix
propagated from time t_i to t_{i+1}

\underline{Q}_f = noise covariance kernel description
as defined in Equation (3-7)

The filter state transition matrix, $\underline{\phi}_f(t_{i+1}, t_i)$, must satisfy the differential equation

$$\dot{\underline{\phi}}_f(t, t_i) = \underline{F}_f \underline{\phi}_f(t, t_i) \quad (3-10)$$

over the interval (t_i, t_{i+1}) , with initial condition

$$\underline{\phi}_f(t_i, t_i) = \underline{I} \text{ (identity matrix)}$$

In this application, the plant matrix \underline{F}_f is constant and the fixed sample period results in a constant state transition matrix, $\underline{\phi}_f(t_{i+1}, t_i)$, for all sample periods. Thus,

$$\underline{\phi}_f(t_{i+1}, t_i) = \begin{bmatrix} 1 & 0 & \Delta t & 0 & A & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & A & 0 & 0 \\ 0 & 0 & 1 & 0 & B & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & B & 0 & 0 \\ 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{df}}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{df}}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{af}}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{af}}} \end{bmatrix} \quad (3-11)$$

where

$$A = T_{df}^2 \begin{bmatrix} \frac{\Delta t}{T_{df}} & -1 + e^{-\frac{\Delta t}{T_{df}}} \end{bmatrix}$$

$$B = T_{df} \begin{bmatrix} 1 - e^{-\frac{\Delta t}{T_{df}}} \end{bmatrix}$$

$$\Delta t = (t_{i+1} - t_i)$$

The solution to the integral term of Equation (3-9), \underline{Q}_{fd} , which represents the growth in uncertainty due to dynamic driving noise since the last measurement update becomes

$$\underline{Q}_{fd} = \begin{bmatrix} Q_{11} & 0 & Q_{13} & 0 & Q_{15} & 0 & 0 & 0 \\ 0 & Q_{11} & 0 & Q_{13} & 0 & Q_{15} & 0 & 0 \\ Q_{13} & 0 & Q_{33} & 0 & Q_{35} & 0 & 0 & 0 \\ 0 & Q_{13} & 0 & Q_{33} & 0 & Q_{35} & 0 & 0 \\ Q_{15} & 0 & Q_{35} & 0 & Q_{55} & 0 & 0 & 0 \\ 0 & Q_{15} & 0 & Q_{35} & 0 & Q_{55} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & Q_{77} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & Q_{77} \end{bmatrix} \quad (3-12)$$

where

$$Q_{11} = \sigma_{df}^2 \left[\frac{(2)(T_{df})(\Delta t)^3 - (2)(T_{df})^2(\Delta t)^2 - (4)(T_{df})^3(\Delta t)(e^{-\frac{\Delta t}{T_{df}}})}{3} + (2)(T_{df})^3(\Delta t) - (T_{df})^4(e^{-\frac{2\Delta t}{T_{df}}}) + T_{df}^4 \right]$$

$$Q_{13} = \sigma_{df}^2 \left[(T_{df})(\Delta t)^2 + (2)(T_{df})^2(\Delta t)(e^{-\frac{\Delta t}{T_{df}}}) + T_{df}^3 - (2)(T_{df})^3(e^{-\frac{\Delta t}{T_{df}}}) - (2)(T_{df})^2(\Delta t) + (T_{df})^3(e^{-\frac{2\Delta t}{T_{df}}}) \right]$$

$$Q_{15} = \sigma_{df}^2 \left[(-2)(T_{df})(\Delta t)(e^{-\frac{\Delta t}{T_{df}}}) + T_{df}^2 - (T_{df})^2(e^{-\frac{2\Delta t}{T_{df}}}) \right]$$

$$Q_{33} = \sigma_{df}^2 \left[(2) (T_{df}) (\Delta t) - (3) (T_{df})^2 + (4) (T_{df})^2 \left(e^{\frac{-\Delta t}{T_{df}}} \right) - (T_{df})^2 \left(e^{\frac{-2\Delta t}{T_{df}}} \right) \right]$$

$$Q_{35} = \sigma_{df}^2 \left[T_{df} - (2) (T_{df}) \left(e^{\frac{-\Delta t}{T_{df}}} \right) + (T_{df}) \left(e^{\frac{-2\Delta t}{T_{df}}} \right) \right]$$

$$Q_{55} = \sigma_{df}^2 \left[1 - e^{\frac{-2\Delta t}{T_{df}}} \right]$$

$$Q_{77} = \sigma_{df}^2 \left[1 - e^{\frac{-2\Delta t}{T_{af}}} \right]$$

Note that $Q_{22} = Q_{11}$, $Q_{44} = Q_{33}$, $Q_{66} = Q_{55}$, $Q_{88} = Q_{77}$, $Q_{24} = Q_{13}$, $Q_{26} = Q_{15}$, $Q_{46} = Q_{35}$. For development of the noise covariance matrix, Equation (3-12), see Appendix B.

The matrix of Equation (3-12) is constant for a given sample time and fixed T_{df} and T_{af} . The values of σ_{df}^2 and σ_{af}^2 of the Equation (3-7) can either be determined offline during a filter tuning process or adaptively in real-time although in this instance σ_{af}^2 was always determined offline. This tuning process is used to select the values of these parameters which optimize tracking performance, optimum in the sense that the mean square tracking errors are minimized, for specific scenarios. Equations (3-8) and (3-9) propagate the filter states and conditional covariance forward to the next sample time. Once the predicted states for the next sample time are calculated, the estimates of the target's position in azimuth and elevation are fed back to the controller. The controller then positions the center of the FLIR field-of-view at that predicted location by the next sample time. Since the filter assumes that the FLIR is centered on the true target position, the state estimate prior to the measurement update is

$$\hat{\mathbf{x}}(t_{i+1}^-) = \begin{bmatrix} 0 & 0 & \hat{x}_{vel} & \hat{y}_{vel} & \hat{x}_{acc} & \hat{x}_{atmos} & \hat{y}_{atmos} \end{bmatrix}^T \quad (3-13)$$

i.e., $\hat{\underline{x}}(t_{i+1}^-)$ of Equation (3-8) with the first two components zeroed out.

3.5 Measurement Update Equation

The two-dimensional discrete-time measurement vector generated by the correlation algorithm, $\underline{z}(t_i)$, is a linear combination of the state variables of interest, i.e. the apparent target intensity function's true position corrupted by uncertain measurement disturbance, $\underline{v}(t_i)$. Thus,

$$\underline{z}(t_i) = \underline{H}_f \underline{x}_f(t_i) + \underline{v}_f(t_i) \quad (3-14)$$

where

$$\underline{z}(t_i) = \begin{bmatrix} x_c \\ y_c \end{bmatrix} = \begin{matrix} \text{the estimated x and y coordinates of the} \\ \text{centroid of the target intensity function} \\ \text{as estimated by the correlation algorithm} \end{matrix}$$

$$\underline{H}_f(t_i) = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 1 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 1 \end{bmatrix} = \begin{matrix} \text{linear combination of the} \\ \text{state variables which} \\ \text{contribute to the measurement} \\ \text{elements} \end{matrix}$$

$\underline{v}_f(t_i)$ = additive noise assumed to be white Gaussian with statistics:

$$E[\underline{v}_f(t_i)] = \underline{0}$$

$$E[\underline{v}_f(t_i) \underline{v}_f^T(t_j)] = \underline{R}_f(t_i) \delta_{ij}$$

The measurement information, $\underline{z}(t_i)$, is incorporated by the Kalman filter using the standard update equations:

$$\begin{aligned}
\underline{K}_f(t_i) &= \underline{P}_f(t_i^-) \underline{H}_f^T(t_i) \{ \underline{H}_f(t_i) \underline{P}_f(t_i^-) \underline{H}_f^T(t_i) + \underline{R}_f(t_i) \}^{-1} \\
\underline{\hat{x}}_f(t_i^+) &= \underline{\hat{x}}_f(t_i^-) + \underline{K}_f(t_i) \{ \underline{z}_f(t_i) - \underline{H}_f(t_i) \underline{\hat{x}}_f(t_i^-) \} \\
\underline{P}_f(t_i^+) &= \underline{P}_f(t_i^-) - \underline{K}_f(t_i) \underline{H}_f(t_i) \underline{P}_f(t_i^-)
\end{aligned} \tag{3-15}$$

where

$\underline{\hat{x}}_f(t_i^-), \underline{P}_f(t_i^-)$ = state estimate and conditional covariance obtained from Equations (3-8) and (3-9)

$\underline{K}_f(t_i)$ = Kalman filter gain

$\underline{R}_f(t_i)$ = measurement uncertainty covariance matrix as determined from a statistical analysis of measurement errors (see Chapter 4) and assumed constant for off-line tuning

For a detailed development of the Kalman filter equations, see reference 12.

3.6 \underline{Q}_D Estimation

Online parameter estimation techniques are often used to improve the quality of the state estimates when uncertain parameters exist within the system model or noise covariances. Although a variety of estimation techniques are available (13: Chapter 10), a self-tuning \underline{Q}_D adaptation technique was chosen since all of the information required to implement this estimator is available from the Kalman filter equations, and the computational loading incurred by implementing this method is substantially less than the other estimation techniques referenced.

The estimator is developed as follows:

$$\underline{P}_f(t_i^-) = \underline{\Phi}_f(t_i, t_{i-1}) \underline{P}_f(t_{i-1}^+) \underline{\Phi}_f^T(t_i, t_{i-1}) + \underline{Q}_{fd}(t_{i-1}) \quad (3-16)$$

and

$$\underline{P}_f(t_i^+) = \underline{P}_f(t_i^-) - \underline{K}_f(t_i) \underline{H}_f(t_i) \underline{P}_f(t_i^-) \quad (3-17)$$

Solving Equation (3-16) for $\underline{P}(t_i^-)$ yields

$$\underline{P}_f(t_i^-) = \underline{P}_f(t_i^+) + \underline{K}_f(t_i) \underline{H}_f(t_i) \underline{P}_f(t_i^-) \quad (3-18)$$

where the $\underline{K}_f(t_i) \underline{H}_f(t_i) \underline{P}_f(t_i^-)$ term is preserved intact because it will be estimated on the basis of observed residuals. Setting Equation (3-18) equal to Equation (3-16) and solving for $\underline{Q}_{fd}(t_i)$ yields

$$\begin{aligned} \underline{Q}_{fd}(t_i) = & \underline{K}_f(t_i) \underline{H}_f(t_i) \underline{P}_f(t_i^-) + \underline{P}_f(t_i^+) - \underline{\Phi}_f(t_i, t_{i-1}) \cdot \\ & \cdot \underline{P}_f(t_{i-1}^+) \underline{\Phi}_f^T(t_i, t_{i-1}) \end{aligned} \quad (3-19)$$

The only term in Equation (3-19) not readily available from the filter is the first term. To obtain this information, rewrite the state update equation, Equation (3-15), as

$$\underline{\hat{x}}_f(t_i^+) - \underline{\hat{x}}_f(t_i^-) = \underline{K}_f(t_i) \underline{r}_f(t_i) = \Delta \underline{x}(t_i) \quad (3-20)$$

An estimate of $E\{\Delta \underline{x}(t_i) \Delta \underline{x}^T(t_i)\}$ may be obtained by invoking ergodicity to replace the ensemble average with a single sample time average:

$$E\{\Delta \underline{x}(t_i) \Delta \underline{x}^T(t_i)\} \triangleq \frac{1}{N} \sum_{j=i-N+1}^i \{\Delta \underline{x}(t_j) \Delta \underline{x}^T(t_j)\} \quad (3-21)$$

Assuming steady state filter operation, the residual sequence, $\Delta \underline{x}(t_i)$, can be shown to be a white Gaussian sequence with zero mean and covariance $\underline{K}_f(t_i) \underline{H}_f(t_i) \underline{P}_f(t_i^-)$ (12:229):

$$\underline{K}_f(t_i) \underline{H}_f(t_i) \underline{P}_f(t_i^-) \approx \frac{1}{N} \sum_{j=1-N+1}^i \{ \Delta \underline{x}(t_j) \Delta \underline{x}^T(t_j) \} \quad (3-22)$$

Substituting Equation (3-22) into Equation (3-19) gives $\hat{\underline{Q}}_{fd}(t_i)$ as:

$$\begin{aligned} \hat{\underline{Q}}_{fd}(t_i) = & \frac{1}{N} \sum_{j=1-N+1}^i \{ \Delta \underline{x}(t_j) \Delta \underline{x}^T(t_j) \} + \underline{P}_f(t_i^+) \\ & - \underline{\phi}_f(t_i, t_{i-1}) \underline{P}_f(t_{i-1}^+) \underline{\phi}_f^T(t_i, t_{i-1}) \end{aligned} \quad (3-23)$$

However, rather than averaging over just the first term, this can be approximated by averaging the entire relation over the most recent N sample periods:

$$\begin{aligned} \hat{\underline{Q}}_{fd}(t_i) = & \frac{1}{N} \sum_{j=i-N+1}^i \left[\{ \Delta \underline{x}(t_j) \Delta \underline{x}^T(t_j) \} + \underline{P}_f(t_j^+) \right. \\ & \left. - \underline{\phi}_f(t_j, t_{j-1}) \underline{P}_f(t_{j-1}^+) \underline{\phi}_f^T(t_j, t_{j-1}) \right] \end{aligned} \quad (3-24)$$

This can also be viewed as an approximated maximum likelihood estimate of \underline{Q}_d to be achieved simultaneously with state estimation (13:Chapter 10).

To reduce data storage requirements, a fading memory approximation to finite memory averaging was employed (6:86):

$$\hat{\underline{Q}}_{fd}(t_i) = \hat{\underline{Q}}_{fd}(t_{i-1}) + (1 - \alpha) \underline{Q}_{fd1}(t_i) \quad (3-25)$$

where

$\underline{Q}_{fd1}(t_i)$ = the single summation term in (3-24)
corresponding to time $t_j = t_i$

α = a parameter which essentially controls
how long old estimates of $\hat{\underline{Q}}_{fd}$ are maintained,
i.e., the length of the effective memory
in the fading memory

In Equation (3-24), setting $\alpha = 1$ would use only the
old $\hat{\underline{Q}}_{fd}(t_{i-1})$ and ignore the current data, while setting
 $\alpha = 0$ would result in ignoring all previous $\hat{\underline{Q}}_{fd}$ estimates.
Therefore, a low α value implies that little useful
information is believed to be contained in the previous
estimates, while a high α responds slowly to current
estimates. Typical values for α are

$$0.7 < \alpha < 1.0$$

Note this is the same fading memory technique employed
in the "Exponential Smoothing of Data" algorithm shown in
Equation (1-4).

IV. Methods of Correlation

4.1 Introduction

The extended Kalman filter has been shown to perform well in comparison to standard correlation trackers when the filter was provided valid a priori information about the analytic form of the intensity function (8). However, without this a priori information the benefit of implementing a high-dimensional measurement extended Kalman filter is greatly reduced. Additionally, the correlation algorithm is computationally easier to implement. Consequently, this section explores four correlation methods for possible implementation as data preprocessors to generate a two-dimensional FLIR target image centroid "measurement" to the linear Kalman filter described in Chapter 3.

The correlator in this tracking system is used to generate estimates of position within a noise-corrupted data frame using the estimated intensity function, $h(\hat{x}_f(t_i^-), t_i)$, generated by the data processing algorithm as its template. The template has been positioned with $\hat{x}_f(t_i^-)$, as propagated by the Kalman filter from time t_{i-1} . Additionally, controller action is taken to zero out the first two components of $\hat{x}_f(t_i^-)$, i.e. the filter's estimation of where the true target dynamics will place the target at the next sample time, or

$$\hat{x}_f'(t_i^-) = \{0 \ 0 \ \hat{x}_{vel} \ \hat{y}_{vel} \ \hat{x}_{acc} \ \hat{y}_{acc} \ \hat{x}_{atmos} \ \hat{y}_{atmos}\}^T$$

where the components are evaluated at t_i^- . The incoming noise-corrupted measurement is then received at t_i and is correlated with the template to determine the point of maximum resemblance between the two data arrays. The distance from the center of the template to the point

of maximum correlation, as estimated by the correlation process, in the x and y FLIR directions can be considered as an indication of the error in the Kalman filter's estimate, $\hat{x}_f(t_i^-)$, of the target position, or as a pseudomeasurement of this position. Thus, the offset distances in the x_{FLIR} and y_{FLIR} directions are incorporated by the Kalman filter as measurement data to provide a better estimate, $\hat{x}_f(t_i^+)$, of the true target position. The filter propagates the state estimate forward to the next sample time and the template is repositioned at that location with $\hat{x}_f(t_{i+1}^-)$ for correlation with the next measurement. This method will enhance the performance achieved by standard correlation trackers because:

- a) the data processing algorithm, which provides the correlation template, yields a better representation of the target's true intensity function than raw data used by correlators, b) the Kalman filter statistically characterizes known errors in the correlation process to provide a better estimate, and c) the Kalman filter exploits knowledge of target dynamics and atmospheric to position the template for incorporation of the next measurement, whereas this information is not used in standard correlation methods.

In this chapter, several methods of computing the correlation between the two data arrays are considered and a performance analysis of the most promising methods is detailed. This analysis considers the performance of the correlation algorithms against targets exhibiting both single and multiple hot spots.

4.2 Image Resolution

A fundamental requirement for any of the correlation methods under consideration is that the images to be correlated must have the same spatial resolution. In

this research, these two images are the 8x8 FLIR tracking window, and the 24x24 template array. The area of a pixel in the tracking window is the same as the area of a pixel in the template, thus the two images to be correlated have the same spatial resolution, and no image preprocessing is required. The 24x24 template is generated by the data processing algorithm (8), which assumes that the finite data array is one period of an infinitely periodic two-dimensional sequence. Padding the tracking window with 8 rows and columns of zeros creates a 24x24 array which is assumed to be one period by the Discrete Fourier Transform (DFT). This padding insures that the infinite periodicity assumption will not affect the results within the 8x8 tracking window. If the spread parameter of the intensity pattern, σ_g^2 , is such that the height of the target intensity function is approaching zero near the edge of the tracking window, then padding the tracking window with zeros will not adversely affect the results of the data processing algorithm. In the situation where σ_g^2 is such that significant intensity magnitudes exist outside of the 8x8 array, then to pad with zeros would introduce an artificial edge in the intensity function. Thus, padding with zeros is desired when possible as this will accenuate the true intensity function without discarding useful data. However, since for this application a full FLIR frame consists of 300x400 pixels, the 8x8 data array could be padded with noise corrupted data when the spread parameter of the target intensity function is such that there are significant intensity magnitudes outside of the 8x8 tracking window (8:77-78).

4.3 Correlation Methods

a. Direct Method. The direct method is the classical

approach to determining when two signals match. Consider the two functions shown in Figure 22.

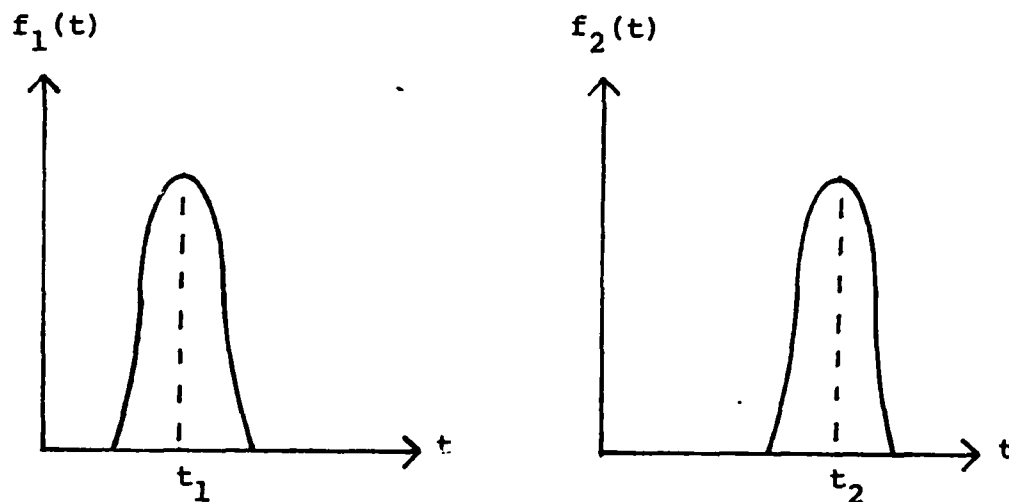


Figure 22. Functions to be Correlated

The correlation integral is defined as

$$C(\tau) = \int_{-\infty}^{\infty} f_1(t) f_2(t+\tau) dt \quad (4-1)$$

Where τ is allowed to take on values from $-\infty$ to $+\infty$.

The correlation peak is defined to be the point where the two signals most closely match, which occurs in Figure 22 at $\tau = t_2 - t_1$.

For discrete signals, Equation (4-1) is approximated by

$$C(p) = T \sum_{n=-\infty}^{\infty} f_1(t) f_2(t+\tau) \quad (4-2)$$

where

$$f_1(t) = f_1(t) \Big|_{t=nT}$$

$$f_2(t) = f_2(t+\tau) \Big|_{\substack{t=nT \\ \tau=pT}}$$

T = sample time

As T becomes small, the result of Equation (4-2) approaches (4-1) and the p which maximizes C(p) is defined as the correlation point. With the two-images to be correlated as shown in Figure 23, the two-dimensional discrete correlation algorithm is (14:17-20):

$$R(p,q) = \frac{1}{KL} \sum_{y=1}^L \sum_{x=1}^K G_1(x,y) G_2(x+p,y+q) \quad (4-3)$$

for

$$0 \leq p \leq X-K$$

$$0 \leq q \leq Y-L$$

(Note: In this instance since the G_1 array is of smaller dimension than the G_2 array negative values of p and q are not required which corresponds to the tracking window being of smaller dimension than the template.)

Since in this research the 8x8 tracking window is padded either with zeros or noise corrupted data, the coordinates of the pixels in the tracking window to be correlated range from 9-16 in the x and y FLIR directions. Equation (4-3) then becomes {14:20}

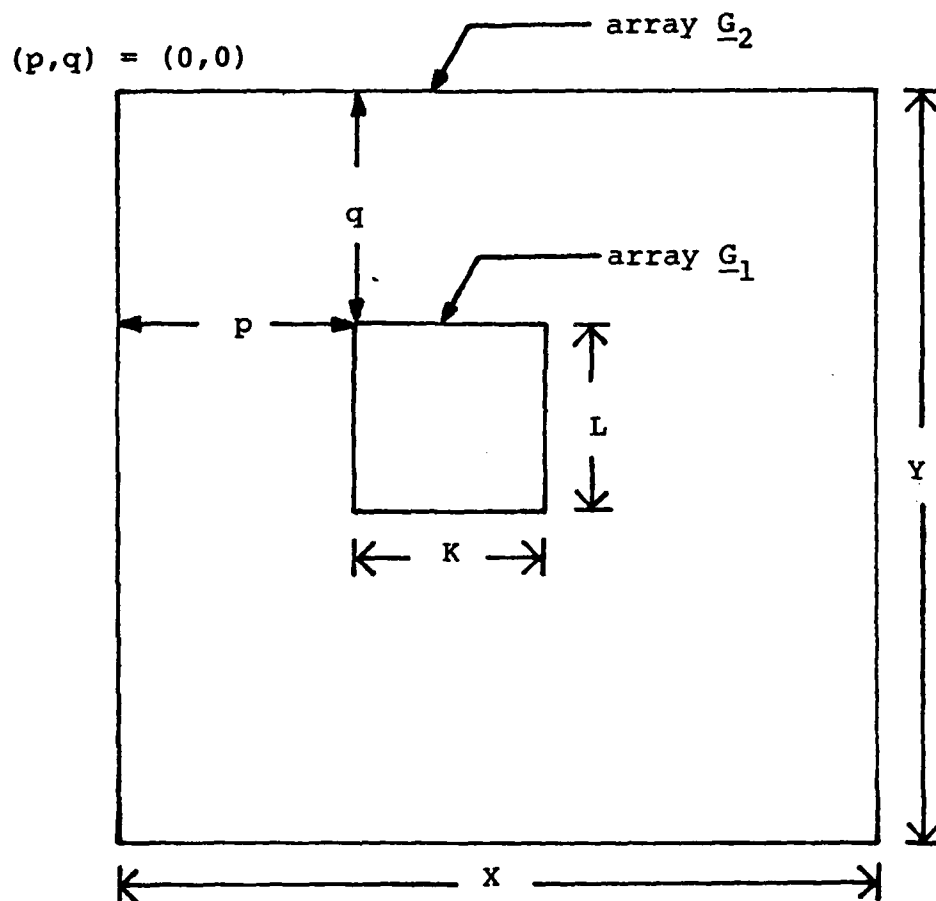


Figure 23. Arrays to be Correlated

$$R(p,q) = \frac{1}{(8)(8)} \sum_{y=1}^8 \sum_{x=1}^8 G_1(x+8,y+8) G_2(x+p,y+q) \quad (4-4)$$

for

$$4 \leq p \leq 12$$

$$4 \leq q \leq 12$$

where

G_1 = 8x8 tracking window within padded data array

G_2 = 24x24 template

This is pictured in Figure 24.

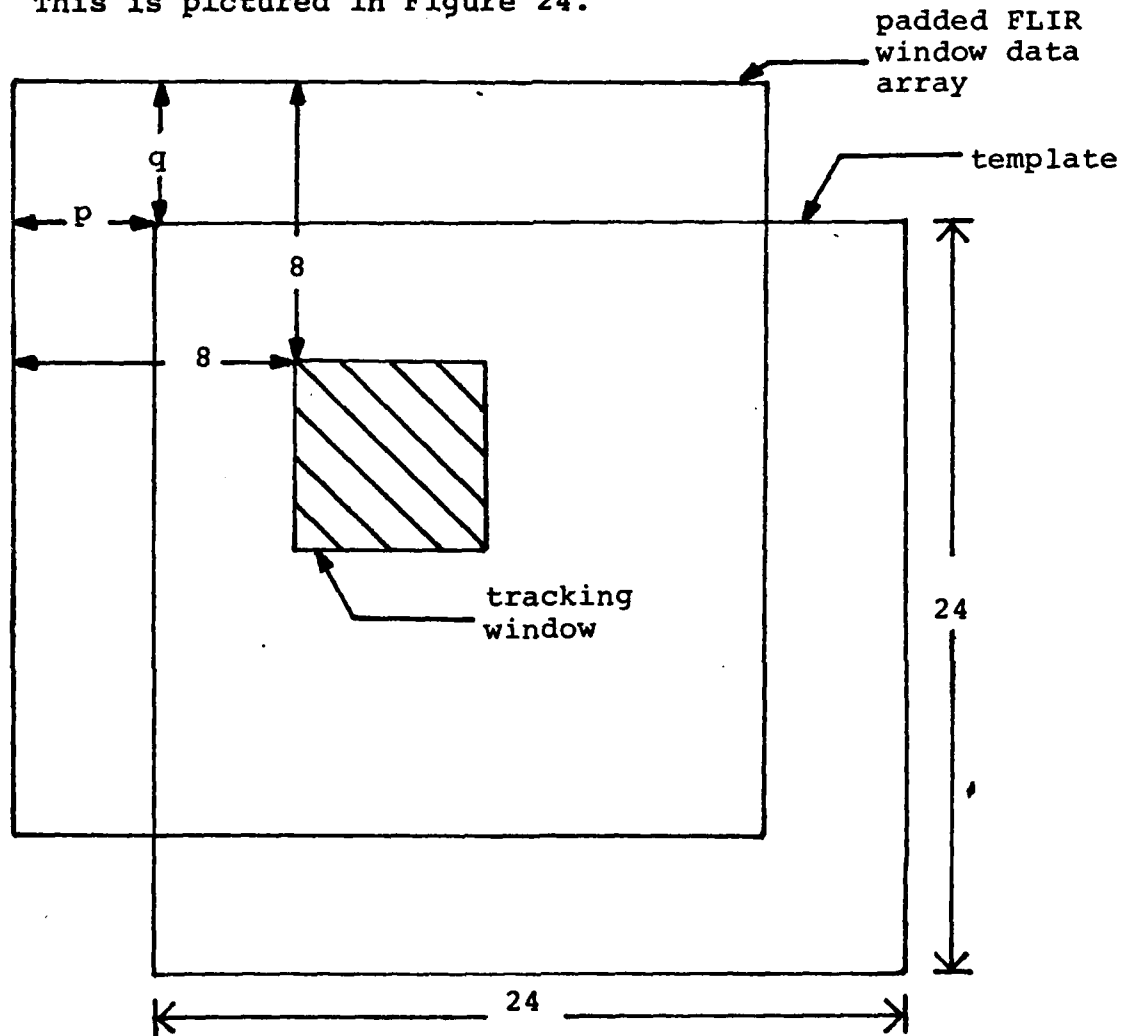


Figure 24. FLIR and Template Arrays

Equation (4-4) is referred to as the direct correlation method (14-20). Notice the range over which p and q are allowed to vary. A total correlation of the two arrays

would have p and q range from 0 to 16. However, if the target intensity function is outside of the 8x8 tracking window no useful measurement information will be contained in the incoming data, and the tracker is considered to have lost the target unless the size of the tracking window is expanded (a possibility not explored in this research). Recall that the template which contains the estimated intensity function offset by the atmospheric states is generated via the data processing algorithm from the measurement arrays which have been padded with 8 rows and columns of either zeros or noise-corrupted data. Therefore, to perform a correlation outside of the p and q region shown is equivalent to correlating the incoming data array with either zeros or smoothed noise. In either case, a correlation match in this area indicates the incoming array contains no useful information, i.e. target track has been lost. For this reason, correlation in the region outside of the chosen p and q bounds was not performed, thereby reducing the computational loading of the algorithm.

Equation (4-4) can be normalized so that the calculation of correlation points is insensitive to magnitude values and depends solely on the pattern of the 8x8 arrays within the template (14:20). The normalized correlation function, $R_N(p,q)$ is:

$$R_N(p,q) = \frac{\sum_{y=1}^8 \sum_{x=1}^8 G_1(x+8,y+8) G_2(x+p,y+q)}{\left[\sum_{y=1}^8 \sum_{x=1}^8 G_1^2(x+8,y+8) \right] \left[\sum_{y=1}^8 \sum_{x=1}^8 G_2^2(x+p,y+q) \right]}^{\frac{1}{2}} \quad (4-5)$$

for

$$4 \leq p \leq 12$$

$$4 \leq q \leq 12$$

Thus, Equation (4-5) defines the normalized direct correlation method implemented in this research.

Actual implementation of Equation (4-5) requires the signals of template and the FLIR data be digitized. The process of digitizing a continuous signal consists of two distinct steps. The first is the sampling of the signal at discrete intervals, the second is quantization. The truth model provides the sampled signal in this instance. Two separate quantizers were implemented to evaluate the merits of using a low-level quantizer which is readily implemented versus the performance enhancement achieved by using a higher-level quantizer, with its associated increased computational burden. The input-output relationship for the two quantizers chosen is shown in Figure 25.

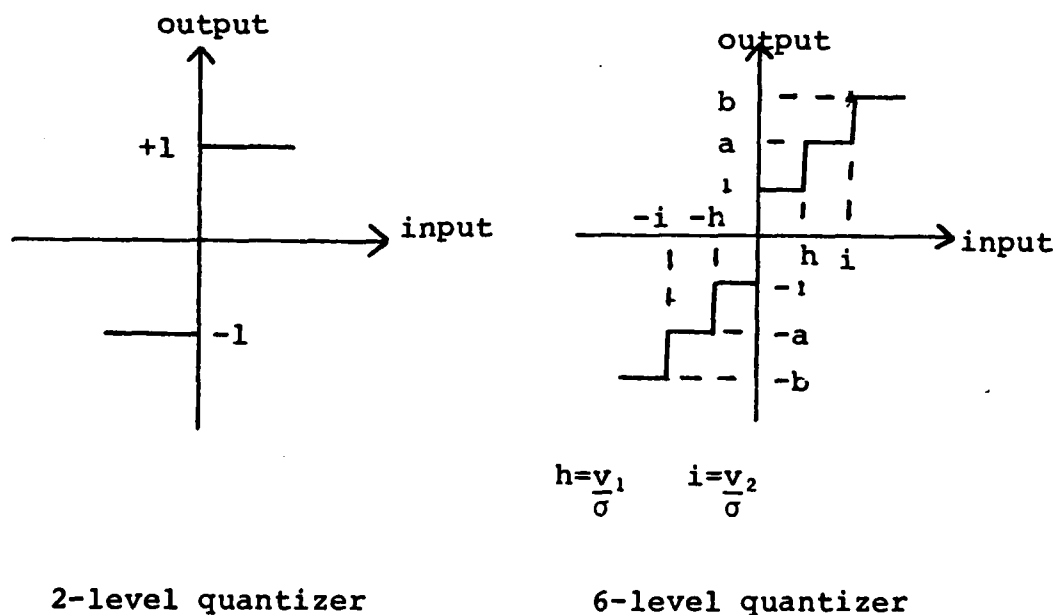


Figure 25. Input-Output Relationships

For the two-level quantizer, a pixel within the array is assigned a value of +1 if the intensity for that pixel is above the mean intensity level, and the pixel is assigned a value of -1 if the signal is below the mean intensity level. The mean intensity values, \bar{G}_1 and $\bar{G}_2(p,q)$ are determined by:

$$\bar{G}_1 = \frac{1}{(8)(8)} \sum_{y=1}^8 \sum_{x=1}^8 G_1(x+8,y+8)$$

$$\bar{G}_2(p,q) = \frac{1}{(8)(8)} \sum_{y=1}^8 \sum_{x=1}^8 G_2(x+p,y+q)$$

Thus, Equation (4-5) becomes (14:32)

$$R_N(p,q) = \frac{\sum_{y=1}^8 \sum_{x=1}^8 g_1(x+8,y+8) g_2(x+p,y+q)}{64} \quad (4-6)$$

for

$$4 \leq p \leq 12$$

$$4 \leq q \leq 12$$

where

$g_1(x+8,y+8)$ = value assigned to $\underline{G}_1(x+8,y+8)$ element after quantization

$g_2(x+p,y+q)$ = value assigned to $\underline{G}_2(x+p,y+q)$ element after quantization

As shown in Figure 25b, the six-level quantization process requires the additional calculation of the variances of the input signal, σ , for $\underline{G}_1(x+8,y+8)$ and $\underline{G}_2(x+p,y+q)$. Optimum values in the sense that they minimize the variance

of R, (i.e. minimize the loss in the signal to noise ratio due to quantization(14:28)), based on research detailed in ref 14 are $a = \pm 3$ and $b = \pm 6$ for $v_1/\sigma = \pm 0.6$ and $v_2/\sigma = \pm 1.4$. Thus, the quantized value, g , is assigned by:

$$\begin{aligned} \frac{v}{\sigma} &\leq -1.4; g = -6 \\ -1.4 &\leq \frac{v}{\sigma} \leq -0.6; g = -3 \\ -0.6 &\leq \frac{v}{\sigma} \leq 0; g = -1 \\ 0 &\leq \frac{v}{\sigma} \leq 0.6; g = 1 \\ 0 &\leq \frac{v}{\sigma} \leq 1.4; g = 3 \\ 1.4 &\leq \frac{v}{\sigma}; g = 6 \end{aligned}$$

where

v = (Intensity value for a particular pixel-mean intensity value for the data array.)

For the six-level quantizer Equation (4-5) becomes:

$$R_N(p,q) = \sum_{y=1}^8 \sum_{x=1}^8 \{g_1(x+8,y+8)g_2(x+p,y+q)\} \\ \left[\left[\sum_{x=1}^8 \sum_{y=1}^8 g_1^2(x+8,y+8) \right] \left[\sum_{x=1}^8 \sum_{y=1}^8 g_2^2(x+p,y+q) \right] \right]^{\frac{1}{2}} \quad (4-7)$$

for

$$4 \leq p \leq 12$$

$$4 \leq q \leq 12$$

The values of p and q which maximize R_N is the correlation point.

b. Fast Fourier Transform (FFT) Method. Since the arrays to be correlated are two-dimensional, rotating one array 180° and taking a convolution is equivalent to performing a correlation. The convolution has the useful property that its discrete Fourier Transform is a simple product of the Fourier Transform of the two image arrays (15:5). This method substantially reduces the number of steps required to calculate the correlation. The FFT can be used to calculate the cross-correlation as shown below.

$$\text{FFT}\{\underline{G}_1(x,y)\} = \underline{G}_1\{f_x, f_y\} \quad (4-8)$$

$$\text{FFT}\{\underline{G}_2(x,y)\} = \underline{G}_2\{f_x, f_y\} \quad (4-9)$$

$$\text{FFT}\{\underline{G}_1(x,y) \cdot \underline{G}_2(x,y)\} = \underline{G}_1\{f_x, f_y\} \cdot \underline{G}_2^*\{f_x, f_y\} \quad (4-10)$$

where

$$\underline{G}_1(x,y) \cdot \underline{G}_2(x,y) = \text{cross-correlation of two-dimensional spatial sequences } \underline{G}_1(x,y) \text{ and } \underline{G}_2(x,y)$$

$$\underline{G}_2^*\{f_x, f_y\} = \text{complex conjugate of Fourier transform of sequence } \underline{G}_2(x,y)$$

FFT = Fast Fourier Transform

By taking the inverse FFT, FFT^{-1} , of Equation (4-10), the cross-correlation of the two-dimensional sequences $\underline{G}_1(x,y)$ and $\underline{G}_2(x,y)$ is obtained (8:53,54)

$$R(x,y) = \underline{G}_1(x,y) \cdot \underline{G}_2(x,y) = \text{FFT}^{-1}\{\underline{G}_1\{f_x, f_y\} \cdot \underline{G}_2^*\{f_x, f_y\}\} \quad (4-11)$$

where

$R(x,y)$ is the result of the correlation

c. Phase Correlation Method. While the FFT method is the simplest and most readily implemented correlation technique, it possesses the undesirable characteristic of detecting false peaks, especially if the FLIR image is of high brightness while the template is of lesser brightness. The problem stems from the relative magnitude of the intensity of the sub-elements (14:22).

The phase correlation method, which reduces the false peak effects of the cross-correlation, is given by (14:22):

$$R'_N(x,y) = \text{FFT}^{-1} \left[\frac{\underline{G}_1(f_x, f_y) \cdot \underline{G}_2^*(f_x, f_y)}{[\underline{G}_1(f_x, f_y) \cdot \underline{G}_2^*(f_x, f_y)]} \right] \quad (4-12)$$

where

$$[\underline{G}_1(f_x, f_y) \cdot \underline{G}_2^*(f_x, f_y)] = \text{magnitude of } \underline{G}_1(f_x, f_y) \cdot \underline{G}_2^*(f_x, f_y)$$

$R'_N(x,y)$ = result of normalized correlation process

This is essentially a point by point normalization in the frequency domain. Due to the normalization, the magnitude effects which lead to the false peaks using the FFT method are reduced and more dependence is placed on the pattern of the data points within the two arrays. However, some of the computational advantage realized by using the FFT method will be lost. The computational loading incurred by using this method will be examined during the performance analysis.

d. Sequential Similiarity Detection Algorithm (SSDA).

Another method considered was the Sequential Similiarity Detection Algorithm. In this method, the absolute value of the difference between corresponding pixels of \underline{G}_1 and \underline{G}_2 , for each p and q, is calculated and the result is defined to be the SSDA registration surface shown in Equation (4-13).

$$E\{p,q\} = \sum_{x=1}^K \sum_{y=1}^L |\underline{G}_1(x,y) - \underline{G}_2(x+p,y+q)| \quad (4-13)$$

for

$$0 \leq p \leq X \leq K$$

$$0 \leq q \leq Y \leq L$$

where

$$|\underline{G}_1(x,y) - \underline{G}_2(x+p,y+q)| = \text{absolute value of the difference}$$

The registration point, which corresponds to the correlation point in the previous methods, is defined as the point for which $E(p,q)$ is a minimum. Note this is not a true correlation technique since Equation (4-13) is not directly related to Equation (4-1). This method was not implemented as analysis has shown that two-level quantization of the SSDA method is equivalent to two-level quantization using the direct method, and for all higher quantization levels, the direct method out performs the SSDA (14:46). However, for actual implementation of a two-level quantizer, hardware considerations may cause the designer to select this method.

4.4 Maximum Correlation Detection

The magnitude of the components of the sequence $R(x,y)$ is a measure of the degree of resemblance between the template and the FLIR image data where the peak location is used to indicate the amount of offset between the two arrays that result in the highest degree of resemblance. Thresholding is a technique often used to suppress lower peaks in the correlation. (Note: In the direct method, the quantization process accomplishes essentially the same function thus thresholding was only employed for the frequency domain correlation methods.) During the thresholding procedure, each element of the $R(x,y)$ sequence is examined to determine if the value for that element is below some preselected fraction of the maximum valued element in the sequence. If the element value is below that minimum point, then that element is considered to have poor correlation and is set to zero. The result is that such elements will have no effect on the computed offset between the template and FLIR data. Figures 26 and 27 show the effects of thresholding as a result of setting the threshold value to .3 and .5 of the maximum valued element respectively. These cases were conducted under identical conditions using the FFT method where the data array was offset from the template by +.15 pixels in the x and y FLIR directions. After the thresholding process has edited the results of the correlation, a center of mass calculation was used to determine the peak of correlation between the two arrays, see Equation (4-14). These plots show the errors in the calculated offset between the two arrays for 1000 cases where a zero error in the plot indicates the correct offset was achieved. As shown in Figure 27, a higher threshold locates the true offset more often but false peaks become prominent. Physically this can be explained by examining the correlation function of this method. For demonstration purposes,

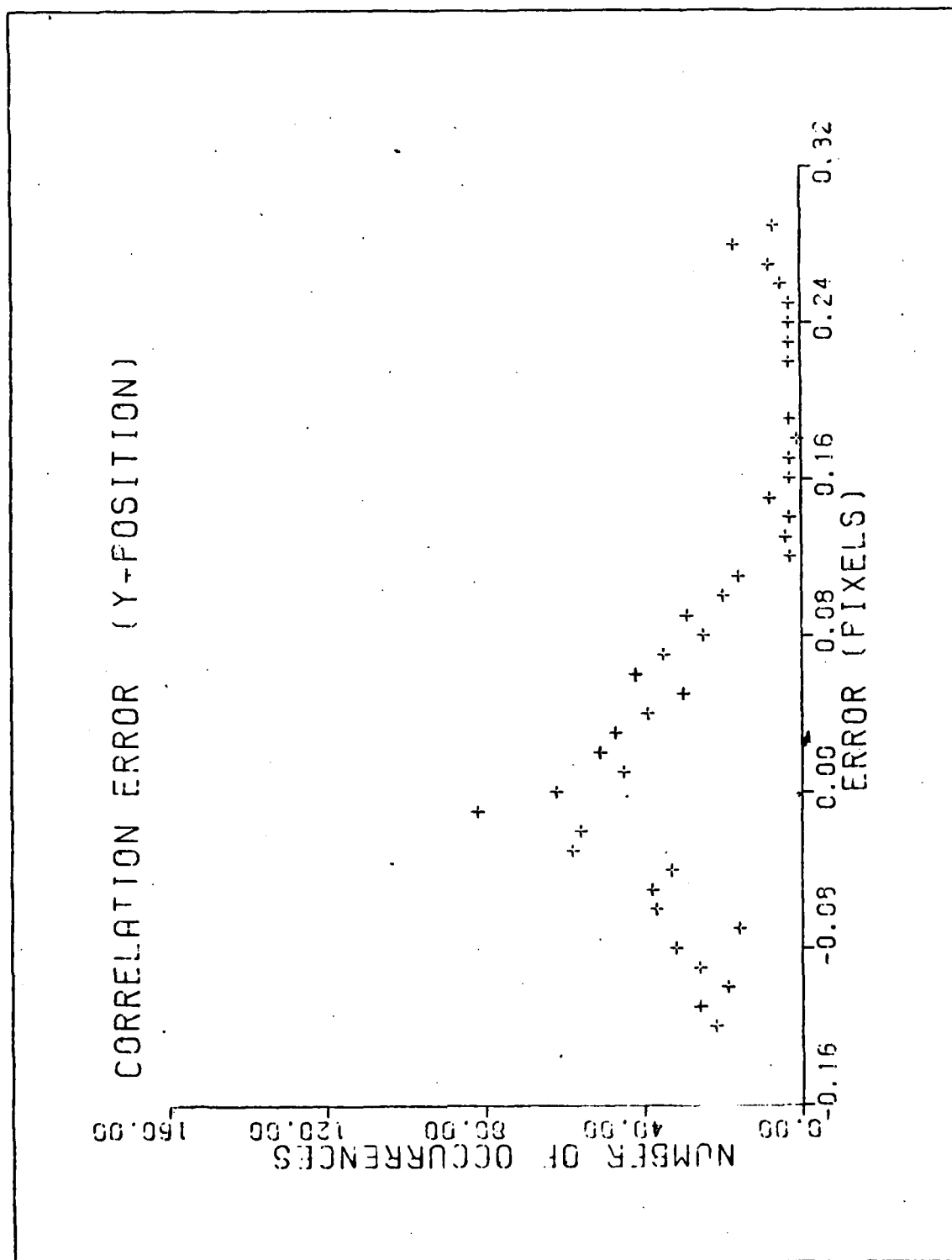


Figure 26. FFT Correlation
(Threshold = .3)

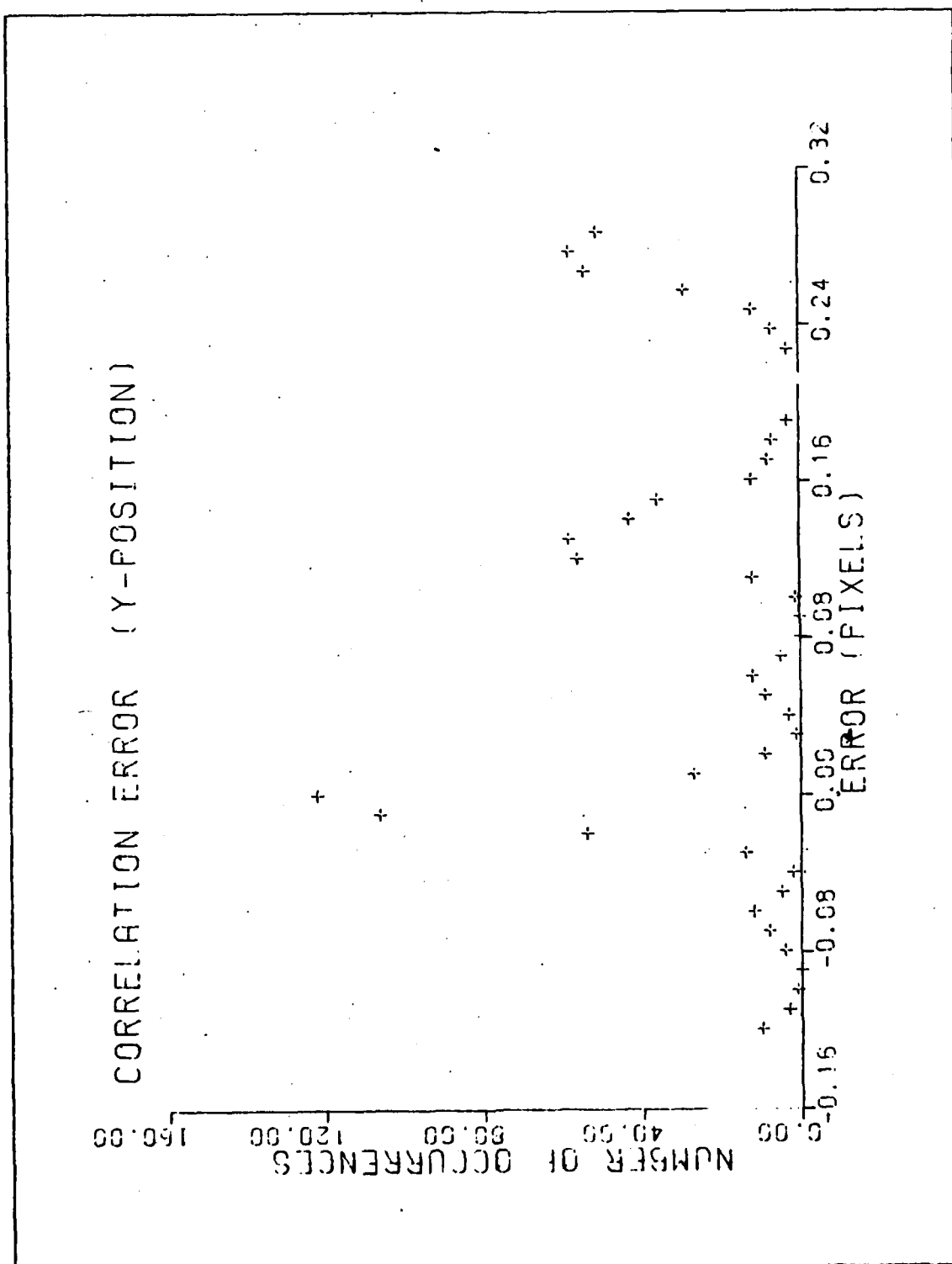


Figure 27. FFT Correlation
(Threshold = .5)

assume Figure 28 is the result of a correlation using the FFT method.

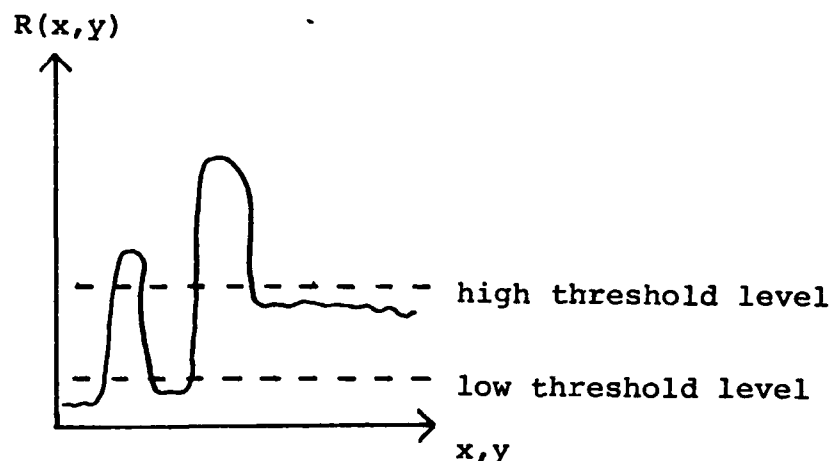


Figure 28. Thresholding (FFT Correlation Method)

As shown, a high degree of correlation lies to the right of the main peak. However, as the threshold is raised, this information is deweighted because the thresholding procedure sets these values to zero and the smaller peak to the left of the main peak becomes weighted more heavily. When the center of mass calculation is used to determine the peak of the correlation function the result will be an estimate which is incorrectly biased to the left. Thus, when using the FFT method the threshold must be set low enough to avoid this effect. However, due to the normalization process the correlation function generated by the phase correlation method would be as

shown in Figure 29.

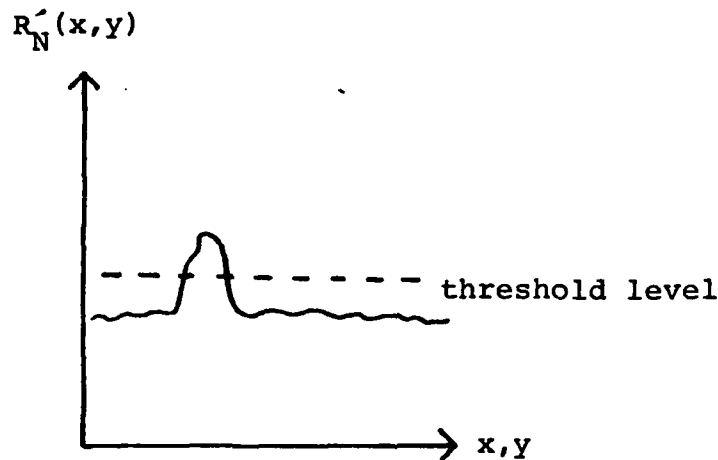


Figure 29. Thresholding (Phase Correlation Method)

Therefore, using this method, a better estimate of the true peak of the correlation function can be obtained by raising the threshold level and then calculating the center of mass.

Once the thresholding method has edited the results of the correlation, the point of maximum correlation must be located. Derivative-based peak detectors, i.e. a method which locates the point of zero slope can be used for this purpose. However, because the location of the point of maximum intensity could be rapidly changing on the FLIR image plane, as when the aircraft being tracked executes a snap roll, it was felt that these type detectors might often misinterpret a local peak as the global peak. Therefore, a centroid summation

AD-A124 884

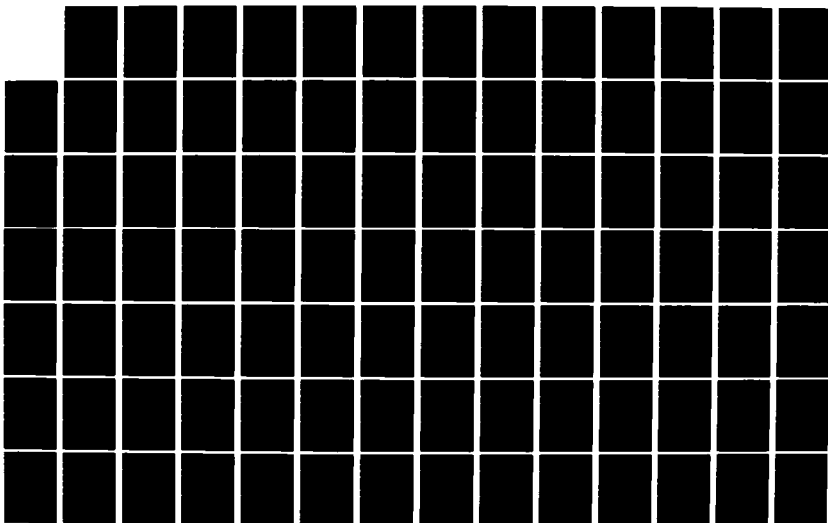
ENHANCED TRACKING OF AIRBORNE TARGETS USING A
CORRELATOR/KALMAN FILTER(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING
P P MILLNER DEC 82 AFIT/GE/EE/82D-58

2/4

UNCLASSIFIED

F/G 12/1

NL



1

technique was selected to locate the center of mass of the edited two-dimensional cross-correlation sequence, $R_T(x,y)$ or $R_N(p,q)$. This method assumes that the center of mass of the thresholded or quantized correlation function is a good indication of the global peak location. The centroid summation used is defined in either the vertical or horizontal direction as (8:63):

$$C = \frac{\sum_{i=1}^N i \cdot \text{Amp}_i}{\sum_{i=1}^N \text{Amp}_i} \quad (4-14)$$

where

i = vertical or horizontal pixel coordinate

Amp = amplitude value for that element

N = total number of pixels in the array

For example, to locate the center of mass of the correlation function in the horizontal direction, the horizontal coordinates of all of the elements of the $R_T(x,y)$ sequence would be multiplied by their respective amplitudes and their products summed. The centroid's horizontal coordinate is then achieved by dividing that summation by the sum of all the amplitudes in the $R_T(x,y)$ sequence. The same procedure is then applied in the vertical direction. The result is that the coordinates of the center of the correlation function in both directions from the center of the template is produced.

4.5 Analysis of Correlation Methods

This section analyzes the errors in the position estimates generated by the four correlation methods previously discussed. The statistical data for the one and three hot spot cases is presented first, followed by a discussion of each methods's performance. The position estimate generated by the correlation algorithm will be the measurements provided to the linear Kalman filter which will provide a better estimate of the target's position. The two-dimensional discrete measurement vector, $\underline{z}(t_i)$, is incorporated by the Kalman filter using the Equation (3-14).

Software was developed to test each of the correlation algorithm's position estimate, and statistical data and histograms of the resulting errors were gathered. For the cases to be discussed the centroid of the intensity was offset from the template by .15 pixels where this distance was selected as being representative of the propagation error. Simulated background and FLIR noise was added as described in section 2.4, and 1000 runs were made to gather the error statistics. As previously described, the errors were calculated so that if the correct offset was estimated a zero error is shown on the histogram, and any deviation from zero is the error in the estimate in pixels. The estimated offset errors were placed into bins .01 pixels wide, and the plots show the number of times the estimated position fell into a particular bin. (The bins were calculated from the mean error to a distance of the mean error \pm .2 pixels where it was assumed the majority of the errors would fall.) Thus, while the area under all the curves has to be 1000 pixels, the total area will not appear on the histogram if the calculated offset was more than .2 pixels from the mean offset error. Table 1 shows the correlator errors for single hot spot targets where the centroid of the intensity function was truly offset from the template by

.15 as previously discussed, and the histograms for these cases are presented in Figures 30-33. (Note the scale in Figure 33 is times 10.)

Table I. Correlation Errors (Single Hot Spot Target)

Correlation Method	\bar{x}_{err}	$\sigma_{x\ err}$	\bar{y}_{err}	$\sigma_{y\ err}$	Time (sec)
Direct 2-level	-.00283	.14995	-.01012	.14875	255.939
Direct 6-level	-.06730	.13697	-.07024	.13510	365.109
FFT Thresh=.3	-.00113	.13262	-.00117	.13374	271.862
Phase Thresh=.7	-.00091	.27046	.00166	.27581	281.344

where

$\bar{x}_{err}, \bar{y}_{err}$ - mean error in the estimated offset in the x and y FLIR directions respectively

$\sigma_{x\ err}, \sigma_{y\ err}$ - standard deviation of the errors in the x and y FLIR directions respectively

Time - total computer simulation time for 1000 runs where all parameters were identical except the correlation method employed

The threshold values used for the FFT and phase correlation

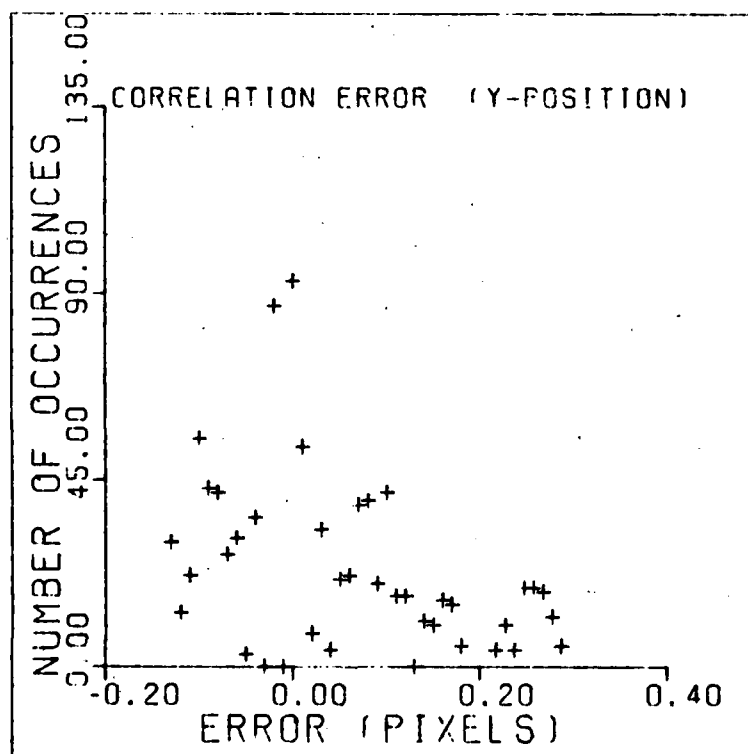
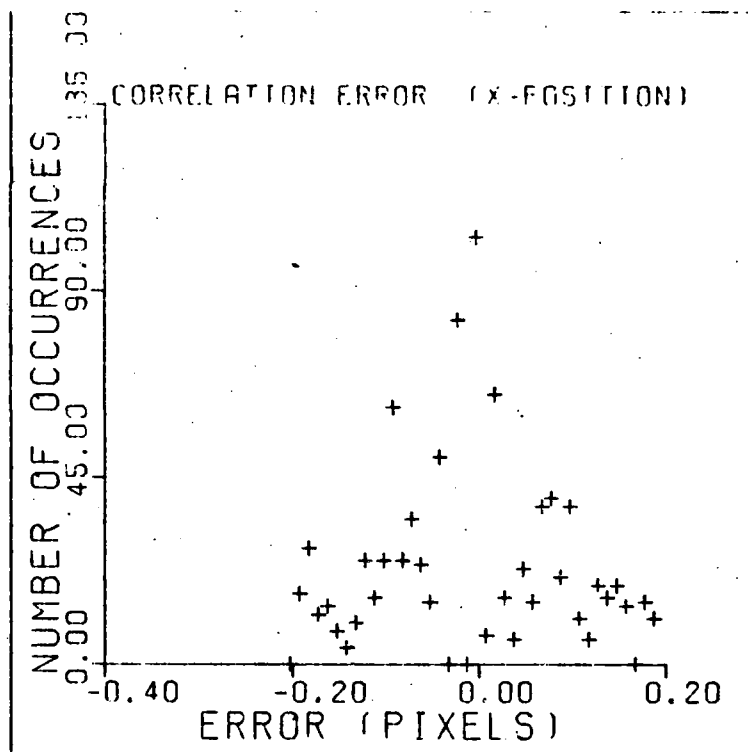


Figure 30. Histogram of Errors-Direct Method
(2-level quantization, 1 hot spot)

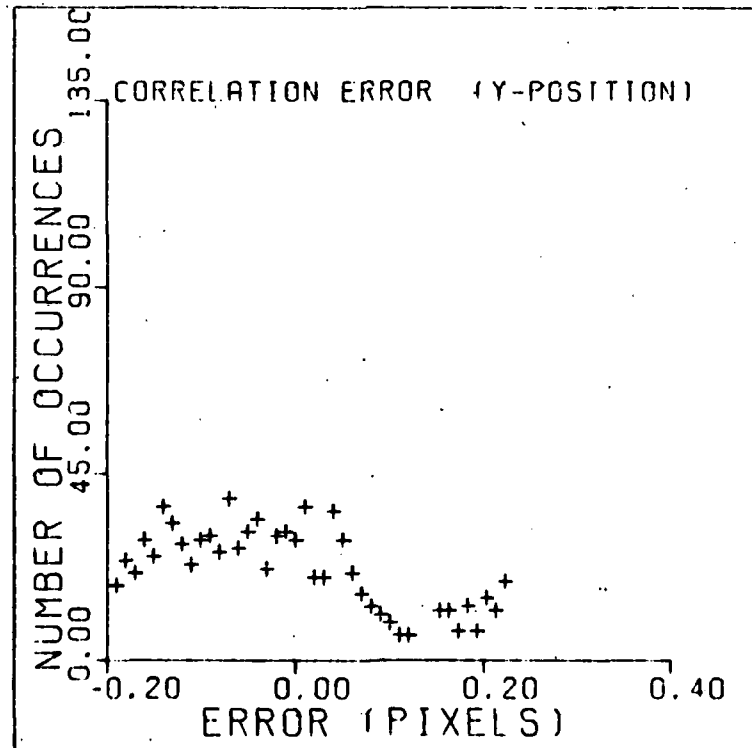
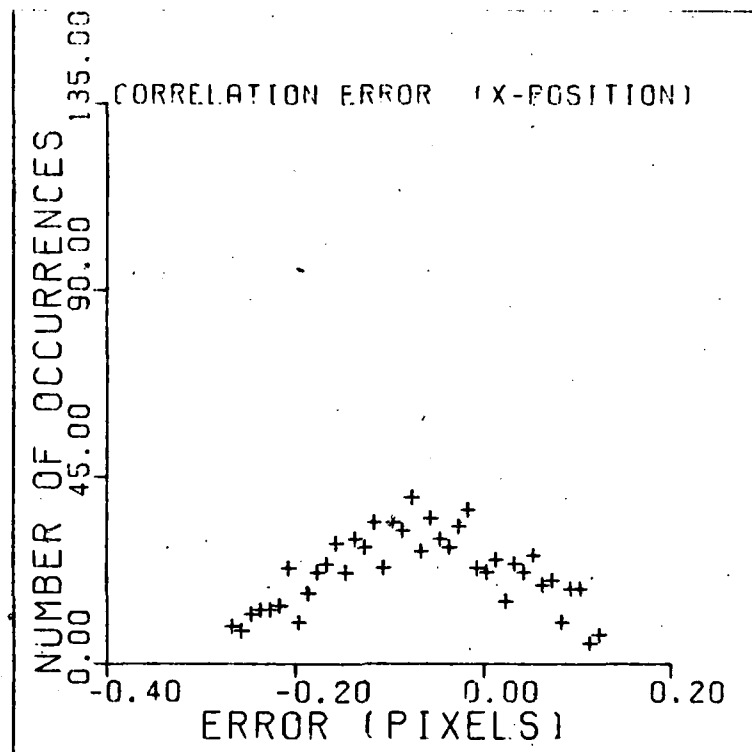


Figure 31. Histogram of Errors-Direct Method
(6-level quantization, 1 hot spot)

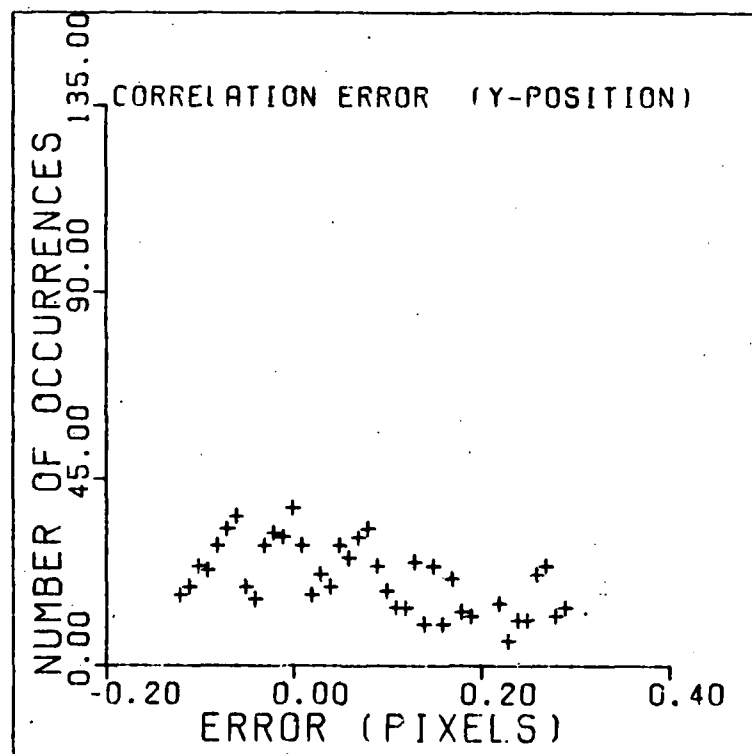
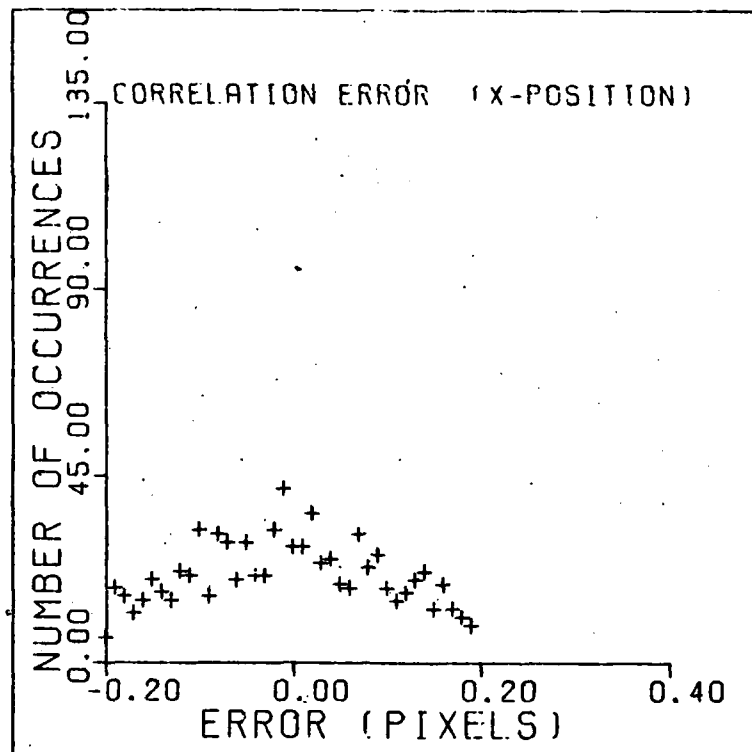


Figure 32. Histogram of Errors-FFT Method
(threshold = .3, 1 hot spot)

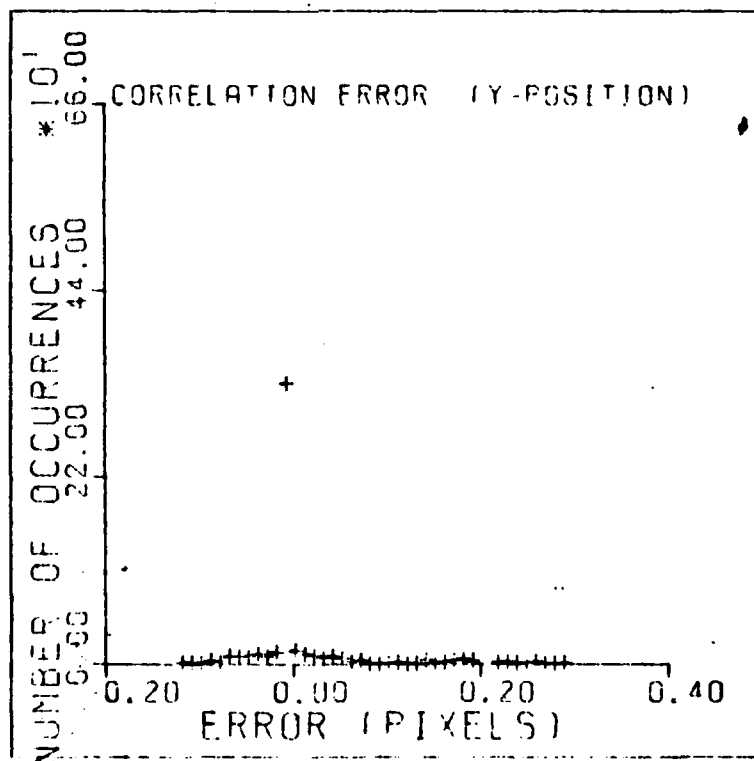
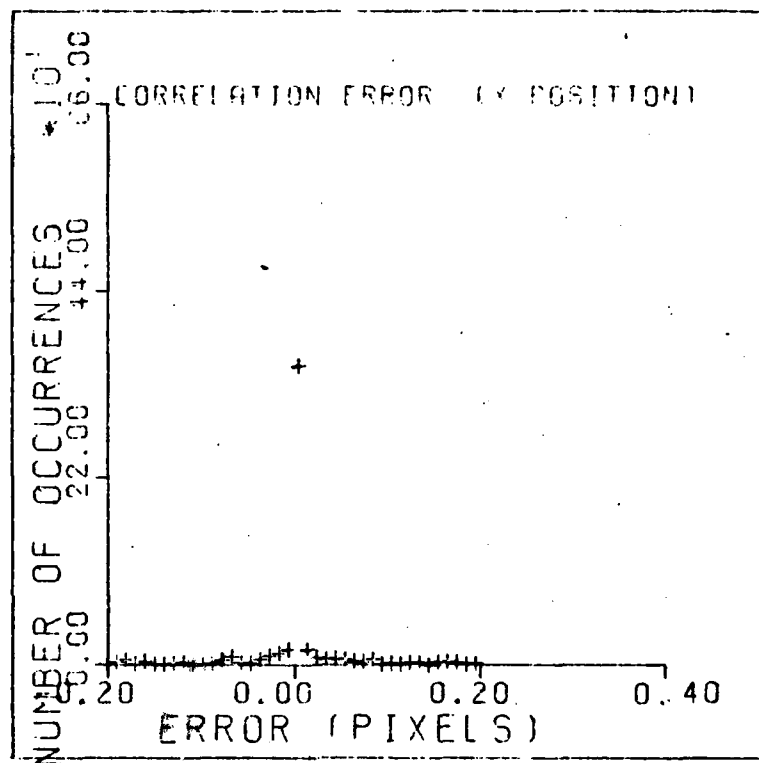


Figure 33. Histogram of Errors-Phase Method
(threshold = .7, 1 hot spot)

methods were chosen as the best values to implement based on a comparison with other trials where various thresholds were used. The threshold value which produced the smallest rms error was chosen.

An error analysis was also conducted for the three hot spot target case. As in the previous simulations, 1000 runs were used to gather the statistical information. The results of the three hot spot cases are shown in Table II.

Table II. Correlation Errors (Three Hot Spot Target)

Correlation Method	\bar{x}_{err}	$\sigma_x \text{ err}$	\bar{y}_{err}	$\sigma_y \text{ err}$	Time (sec)
Direct 2-level	-.03304	.04116	-.01798	.10846	306.268
Direct 6-level	-.08426	.04770	.01754	.03696	400.785
FFT Thresh=.3	-.00053	.03858	.00225	.05428	319.056
Phase Thresh=.7	.00058	.09854	-.00422	.27691	325.461

The histograms of the three hot spot cases are shown in Figures 34-37. (Again, in Figure 37 the scale is times 10.) For the three hot spot cases, the difference in errors in the x and y direction is due to the location of the three target intensity profiles. The centroids of the intensity

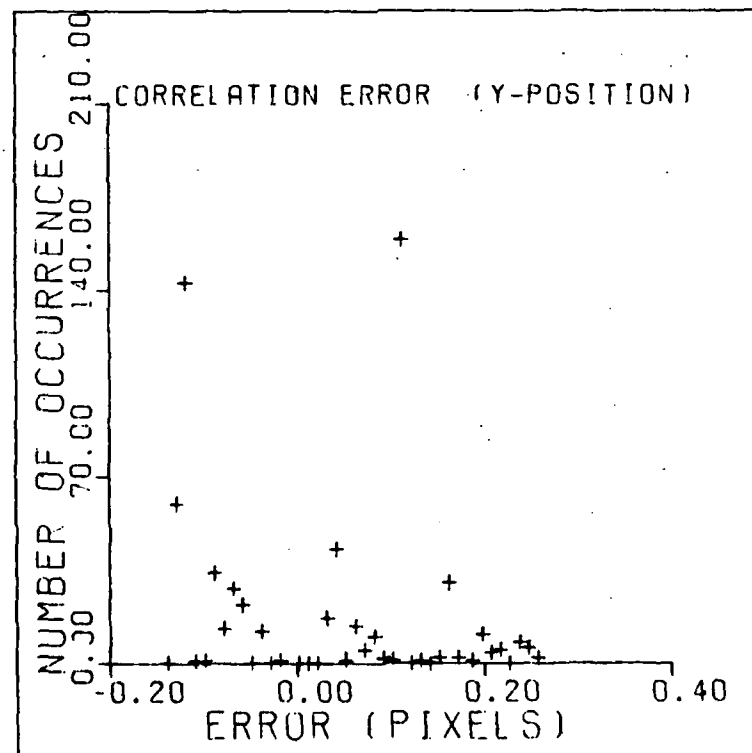
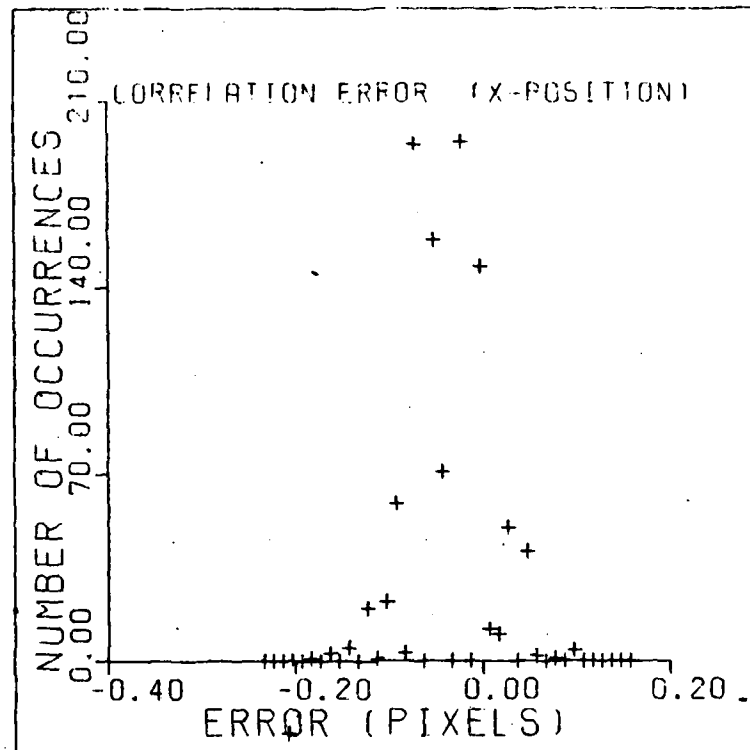


Figure 34. Histogram of Errors-Direct Method
(2-level quantization, 3 hot spots)

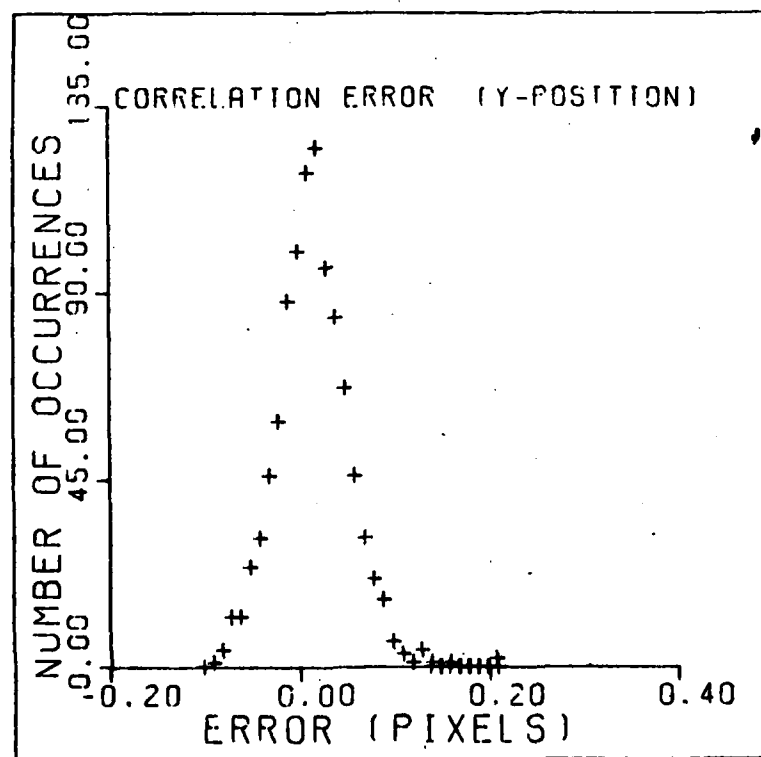
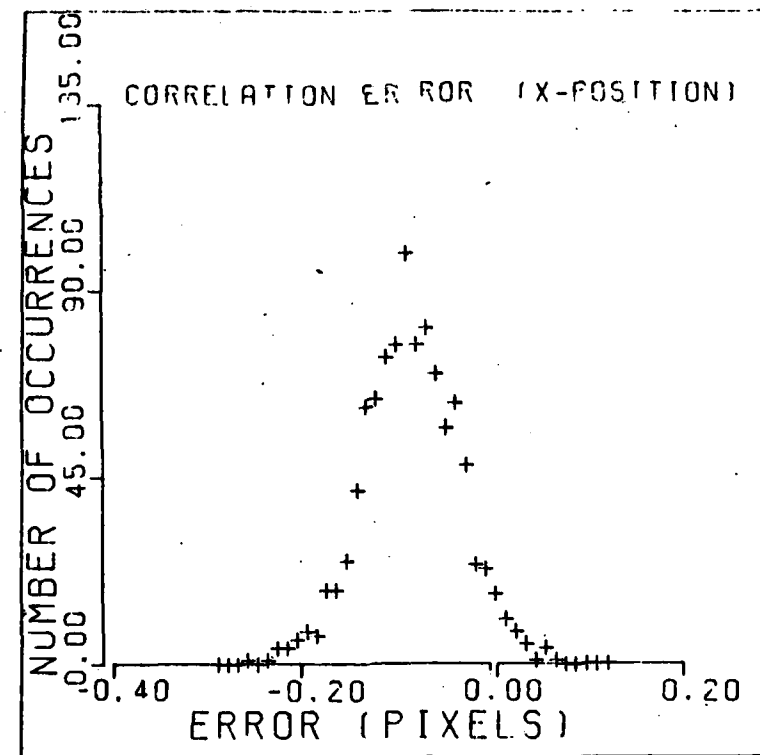


Figure 35. Histogram of Errors-Direct Method
 (-level quantization, 3 hot spots)

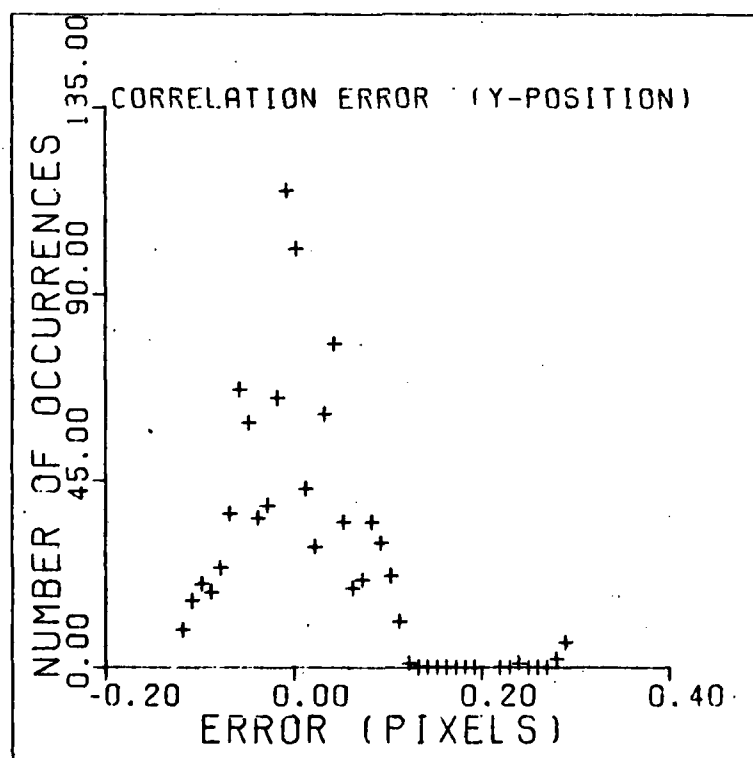
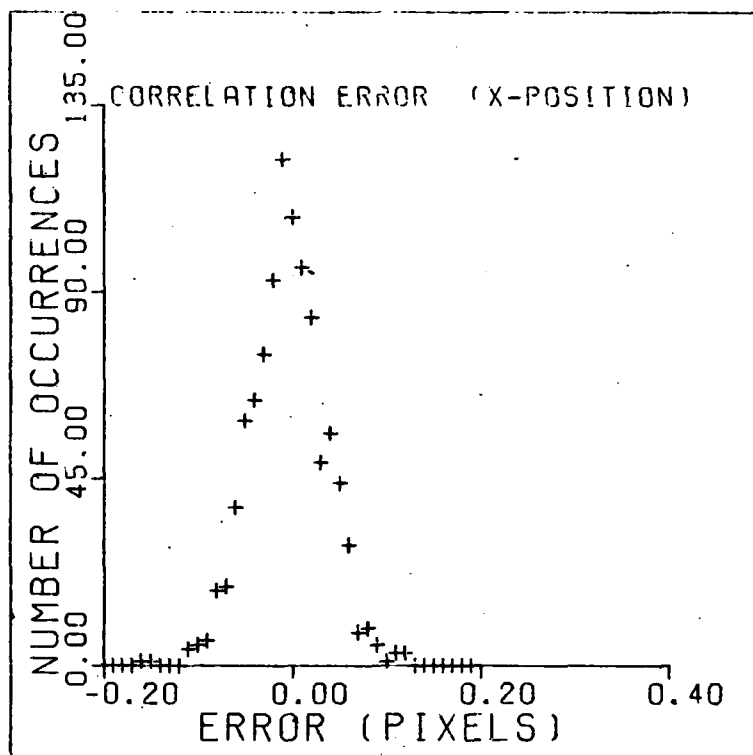


Figure 36. Histogram-FFT Method
(threshold = .3, 3 hot spots)

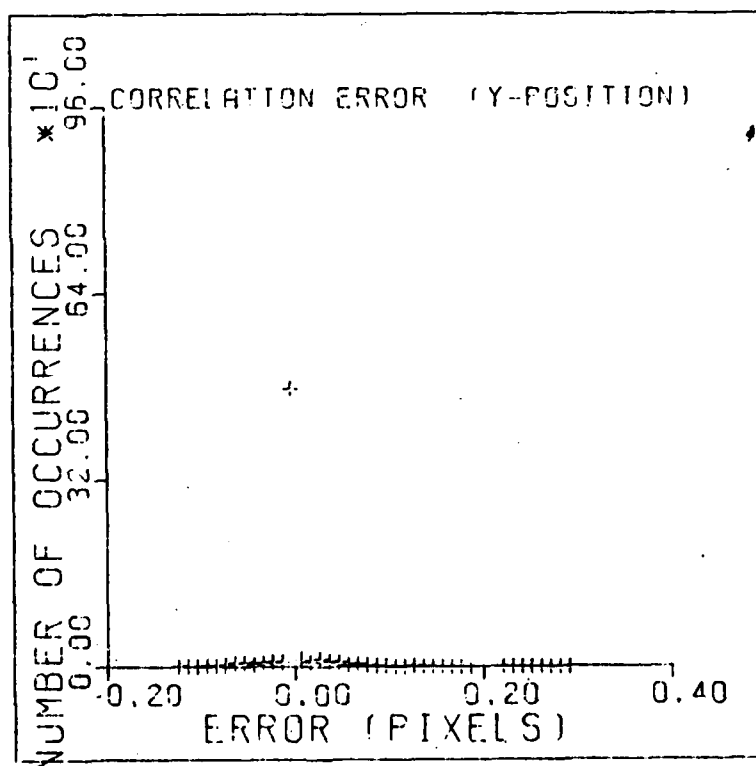
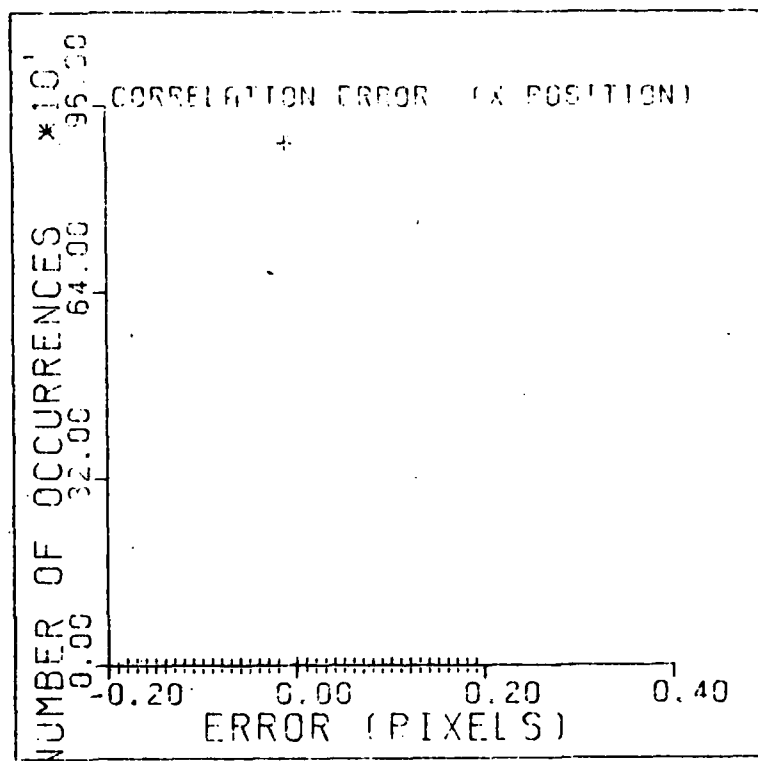


Figure 37. Histogram of Errors-Phase Method
(threshold = .7, 3 hot spots)

profiles were located so that if the center of the 8x8 pixel FLIR tracking window was located at coordinates (0,0) then the target centroids would be at (0,-2.667), (-2,1.33), and (2,1.33). This spacing means the center of mass of the intensity profiles is at (0,0) but the resultant intensity profile is not radially symmetric about (0,0). In order to insure these statistics were representative of the true correlator errors and not of just one particular offset distance chosen, which may have enhanced or degraded the performance of a particular correlation method, evaluations were conducted for other offset distances between the target and the template. The error statistics for these cases, not shown, verified that the cases presented in Tables 1 and 2 were representative of the performance of the correlation methods.

Tables 1 and 2 and Figures (30-37) were analyzed to determine which correlation methods were best suited for implementation with the Kalman filter:

a. Direct Method: (two-level quantization): This is clearly the fastest of the four correlation methods considered. For the one hot spot case its performance is comparable to the other methods with errors which could be modeled as Gaussian. However, the mean error of this method increases from the one to three hot spot case, which is not surprising since this method only categorizes the intensity values as above or below the mean intensity levels. Thus, when there are three hot spots of the same maximum intensity on the FLIR plane, this method has difficulty determining the global peak. This accounts for the erratic nature of the errors in Figure 34. In the x-direction, where the intensity pattern is symmetric, the method performs well. However, in the asymmetric y-direction, there are peaks around 0.0 and ± 1.5 pixels indicating the method is finding the global peak at times and is misinterpreting the peak of

one of three hot spots as the global peak at other times. To model these errors as Gaussian would be questionable. Based on the performance degradation in the three hot spot case, this method was not implemented. However, if computational loading is the driving factor, this method could become more attractive.

b. Direct Method (six-level quantization): The performance of this method is comparable to FFT and phase correlation method with errors which are well modeled as Gaussian. The correlator performs extremely well in determining the peak in the asymmetric y-direction in the three hot spot case. However, the computational loading associated with this method, while gaining no real performance enhancement over the FFT and phase correlation methods, is too great for this method to be considered for implementation.

c. FFT Method (Threshold = .3): The threshold was set low while using this method to avoid the appearance of false peaks which lead to a biased estimate (Figure 27). This correlator performs well with errors, in both the single and multiple hot spot cases, which are well modeled as Gaussian, and in almost all instances, has the smallest standard deviation. The results of this analysis are consistent with the analyses performed by Rogers (8: Chapter V) and this method was implemented with the Kalman filter.

d. Phase Correlation Method (Threshold = .7): As expected, this method substantially reduced the biasing which can occur in the FFT method with the histograms appearing as a spike. However, the computational increase is not substantial. Noting the large standard deviation of this method, the errors committed cannot be well represented as Gaussian which may potentially cause problems. However, because this method does consistently

locate the true global peak of the intensity function, it was implemented with the Kalman filter and a comparison of the tracking performance of this correlation method and the FFT method is presented in Chapter 5.

V. Performance Analysis

5.1 Introduction

This chapter presents the tracking performance of the correlator/Kalman filter when evaluated against the trajectories described in Chapter 2. The first section of the chapter gives an explanation of how the statistics used to evaluate the tracker's performance are computed. The next section of the chapter discusses those parameters within the computer simulation which were changed to evaluate the tracker's performance under various conditions. The third section of the chapter discusses those parameters within the correlator/Kalman filter which can be adjusted off-line in order to enhance the tracking ability of the filter. The final section of the chapter condenses the results of the tracking performance into tables. The statistical information of this chapter was generated using Monte Carlo techniques (12:329). Based on previous variance convergence analyses (6,7), 10 Monte Carlo runs were determined to be sufficient to generate sample statistics that are representative of the true process statistics, and therefore, 10 runs constitute one Monte Carlo study. Each single run consisted of a 5 second, or 150 sample period, simulation.

5.2 Tracking Ability

The errors of primary interest, with respect to the tracking ability of the filter, are errors in the estimated values of $\hat{x}_d(t_i^-)$, $\hat{y}_d(t_i^-)$, $\hat{x}_d(t_i^+)$, and $\hat{y}_d(t_i^+)$. Additionally, since the propagated estimate of the intensity centroid's location, $x_{\text{peak}}(t_{i+1})$ and $y_{\text{peak}}(t_{i+1})$, will affect the correlation process, it is important to estimate this position accurately. The mean error in the

correlator/Kalman filter's estimate for the x-dynamic position can be calculated at any time t_i by

$$\bar{E}_{x_d}(t_i) = \frac{1}{N} \sum_{k=1}^N \{\hat{x}_{df_k}(t_i) - x_{dt_k}(t_i)\} = \frac{1}{N} \sum_{k=1}^N e_{xd_k}(t_i) \quad (5-1)$$

where

$\bar{E}_{x_d}(t_i)$ = mean error (i.e. ensemble average error over all simulations) in the x-dynamic position at time t_i

$\hat{x}_{df_k}(t_i)$ = filter estimated x-dynamics value at time t_i for simulation k

$x_{dt_k}(t_i)$ = truth model, x-dynamics value at time t_i for simulation k

N = number of Monte Carlo runs

and the variance of the error is calculated as

$$\sigma_{xd}^2(t_i) = \frac{1}{N-1} \sum_{k=1}^N e_{xd_k}^2(t_i) - \frac{N}{N-1} \bar{E}_{x_d}^2(t_i) \quad (5-2)$$

Equations (5-1) and (5-2) can be generalized to calculate the errors in the other quantities of interest previously discussed. Additionally, time averages of the mean error and variance were calculated over the last 1.5 seconds of the simulation. This time averaging interval was selected so that any transient effects caused by changes in the target dynamics would have decayed. This provides 45 sample runs, to serve as an indicator of the tracker's performance: time averaging these allows for a compact presentation in tabular form.

5.3 Variation of Truth Model Parameters

Parameters within the truth model are changed to analyze the sensitivity of the correlator/Kalman filter to changes in the "real world". The primary goal of the research was to evaluate the performance of the tracker against targets displaying the various dynamic profiles described in Chapter 2. Thus, the 4 trajectories previously described were the primary truth model factors varied to evaluate the tracker performance. In addition, variations were also considered in parameters defining the target intensity profile itself. To evaluate the performance of the tracker when the strength of the maximum target intensity value changes relative to the background noise, the signal-to-noise ratio (SNR) defined as

$$\frac{S}{N} = \frac{I_{\max}}{\sigma_B} \quad (5-3)$$

where

I_{\max} = maximum target intensity value

σ_B = rms value of background noise, including FLIR noise contributions

is varied. For the multiple hot spot cases, the values of the three Gaussian intensities were equal. Signal to noise ratios of 20 (standard) and 10 were considered as representative of realistic tracking scenarios.

The spread parameter, σ_{pv}^2 , of the target's Gaussian intensity profile(s) can be varied to evaluate the performance of the tracker against sharply peaked (small σ_{pv}^2) or broad (large σ_{pv}^2) Gaussian target intensities. However, because the effects of varying this parameter were previously investigated by Rogers (8:Chapter 6), a

standard value of 2.0 was used throughout this research (of course, σ_{pv} changed as a function of range to the target within a given simulation). However, the target aspect ratio (AR)

$$AR = \frac{\sigma_v}{\sigma_{pv}}$$

was varied to evaluate the tracker's performance against targets exhibiting different intensity profiles due to varying the aspect angle.

5.4 Variation of Data Processing Parameters

Several parameters are available within the data processing algorithm, upper path of Figure 1, which can be varied to improve the estimate of the intensity function. The 8x8 FLIR tracking window can be padded with zeros if the intensity spread parameters, σ_v and σ_{pv} , are such that the target intensity height is approaching zero near the edge of the 8x8 tracking window. However, if these parameters are such that significant intensity magnitudes exist outside the 8x8 tracking windows, then padding with zeros would induce an artificial edge (see Chapter 4). Therefore, the algorithm was structured so the 8x8 FLIR window could be padded with noise-corrupted data when necessary. Based on the results of Roger's research (8:Chapter 6), and because in a dynamic tracking environment significant intensity values may exist near the edge of the tracking window, the FLIR tracking window was padded with noise-corrupted data in this study.

Alpha, the relative weighting parameter for the exponential smoothing process is the next parameter which may be varied within the data processing algorithm. Equation (1-4) displays the role of this parameter in the algorithm.

The standard value used in this research was .05. This value, which indicates that the target intensity profile is essentially averaged over the most recent 20 sample periods, was considered appropriate for targets whose intensity projection was relatively constant. However, if the intensity pattern on the FLIR image plane is varying so that significant changes could occur within a 20-sample period interval of time, such as for a target performing a roll maneuver, increased emphasis should be placed on the more recent measurements. The effects of increasing alpha in this situation are explored.

5.5 Variation of Filter Parameters

Design parameters within the Kalman filter structure are varied either off-line during a filter tuning process or adaptively in real-time in an attempt to improve the quality of the state estimates. The parameters in the linear Kalman filter, developed in Chapter 3, which may be varied during the off-line filter tuning process, and the values of which may change for different target trajectories are the target acceleration and atmospheric jitter time constants, τ_{df} and τ_{af} , the discrete-time noise covariance matrix, \underline{Q}_{fd} , the strength of which may be interpreted as a measure of the uncertainty in the dynamics model being used, (i.e. how adequately the assumed model represents the "real world"), the measurement uncertainty covariance matrix, \underline{R} , and the initial filter covariance matrix, $\underline{P}(t_0)$.

Based upon previous research efforts (6,8,10) the assumed correlation time for the atmospheric jitter position was set at .0707 sec. Using this value assumes the effects of repeated poles shown in Equation (2-2) are considered negligible. The variance of the atmospheric jitter process was set at a constant .2 (pixels²). The diagonal terms for the $\underline{R}(t_i)$ matrix were based on the

statistical analysis of the correlation process detailed in Chapter 4. The cross-correlation between the errors in each of the FLIR image plane directions in the correlation process was calculated and proved to be small enough to assume that the correlation position uncertainty estimate in one direction is independent of the uncertainty of the position estimate in the other direction, resulting in a diagonal $\underline{R}(t_i)$. Thus, the parameters which were used to tune the filter to optimize filter tracking performance were τ_{df} and σ_{df}^2 .

For the type of targets being tracked in this research, target acceleration time constants ranging from 1 to 4 secs were used where a lower τ_{df} is more appropriate for a highly dynamic tracking environment. Values of σ_{df}^2 ranged from 150 (pixels²/sec⁵) to 500 (pixels²/sec⁵) depending on the type of trajectory being tracked, where σ_{df}^2 was increased for highly dynamic targets. The wide disparity in the values used for σ_{df}^2 and σ_{af}^2 directly reflects the range of rms accelerations between benign versus harshly maneuvering targets and their relationship with the jitter rms value.

In the filter tuning process, the $\underline{P}(t_0)$ matrix is set to reflect the knowledge of the conditions under which the estimation process is to be initiated. Since in realistic scenarios, the filter may very well receive inaccurate handoff information from the target acquisition source (such as a multi-target search radar), the diagonal entries of $\underline{P}(t_0)$ were purposefully set high to reflect this handoff uncertainty. The appropriate values at which to initialize variances on the main diagonal were determined by observing the peak values of each of these terms during the filter transient period. The variances on the main diagonal, for position, velocity, and acceleration, were 10 (pixels²), 2000 (pixels²/sec²), and 100 (pixels²/sec⁴) respectively. The initial values of the $\underline{P}(t_0)$ matrix will

be the dominant factor in the initial transient characteristics of the filter (12:337). When tuning the filter, a useful technique is to compare the value of the actual rms errors committed by the filter to the filter's own representation of the error covariance. The time histories of these errors are available as the output of the Monte Carlo simulation process used. By "matching" the filter's rms estimation of its errors to the true rms errors committed by the filter, an attempt is made to insure that the proper relative weighting is given to the internal dynamics model and the measurements (12:338-339).

5.6 Plotting Results

An explanation of the plots generated to depict the filter's tracking errors are given in this section, with the plots which served as the baseline case for the correlator/Kalman filter being shown. The remainder of the performance plots are given in Appendix C. For each run, ten plots were generated. The first two plots show the filter-computed rms errors versus the actual rms errors in the x and y dynamics positions. The next four plots show the mean errors in the x and y dynamics position estimates \pm one standard deviation, beginning with the first plot showing the filter's estimate at time minus and the second plot showing the filter's estimate at time plus. Similarly, the next four plots give the errors in the filter's estimates of the centroid position at times minus and plus. The case number on each plot is used to match each run with the tables shown in the next section. By referring to these tables, the trajectory, truth model parameters, and filter parameters can be cross-referenced. Additionally, a summary of the parameter values for each case is given at the end of the chapter.

Referring to Figure 38a, it is seen that the filter was intentionally set to overestimate its own errors,

FILTER VS ACTUAL ERROR (X-POSITION)

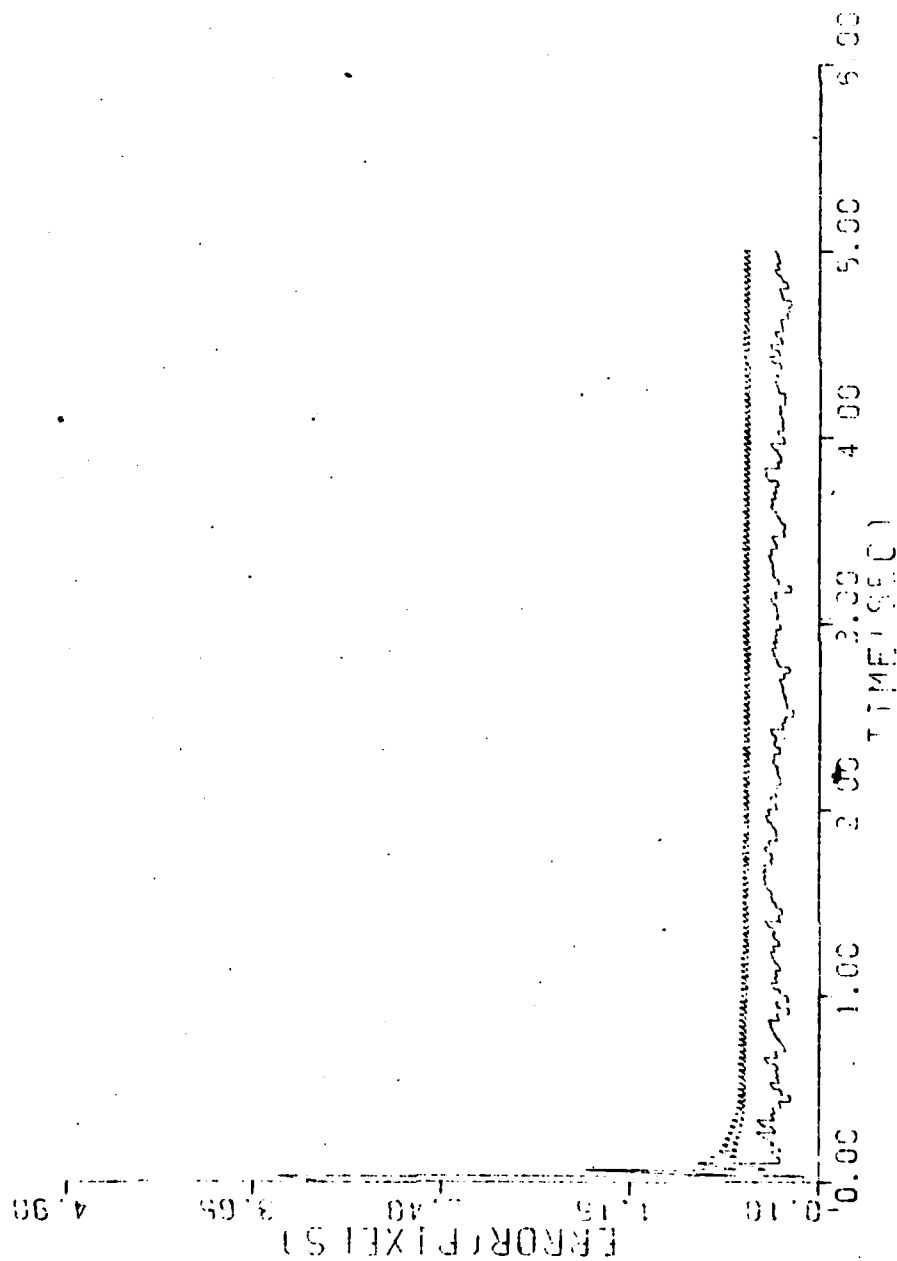


Figure 38a. Case 1

69

FILTER VS ACTUAL ERROR (Y-POSITION)

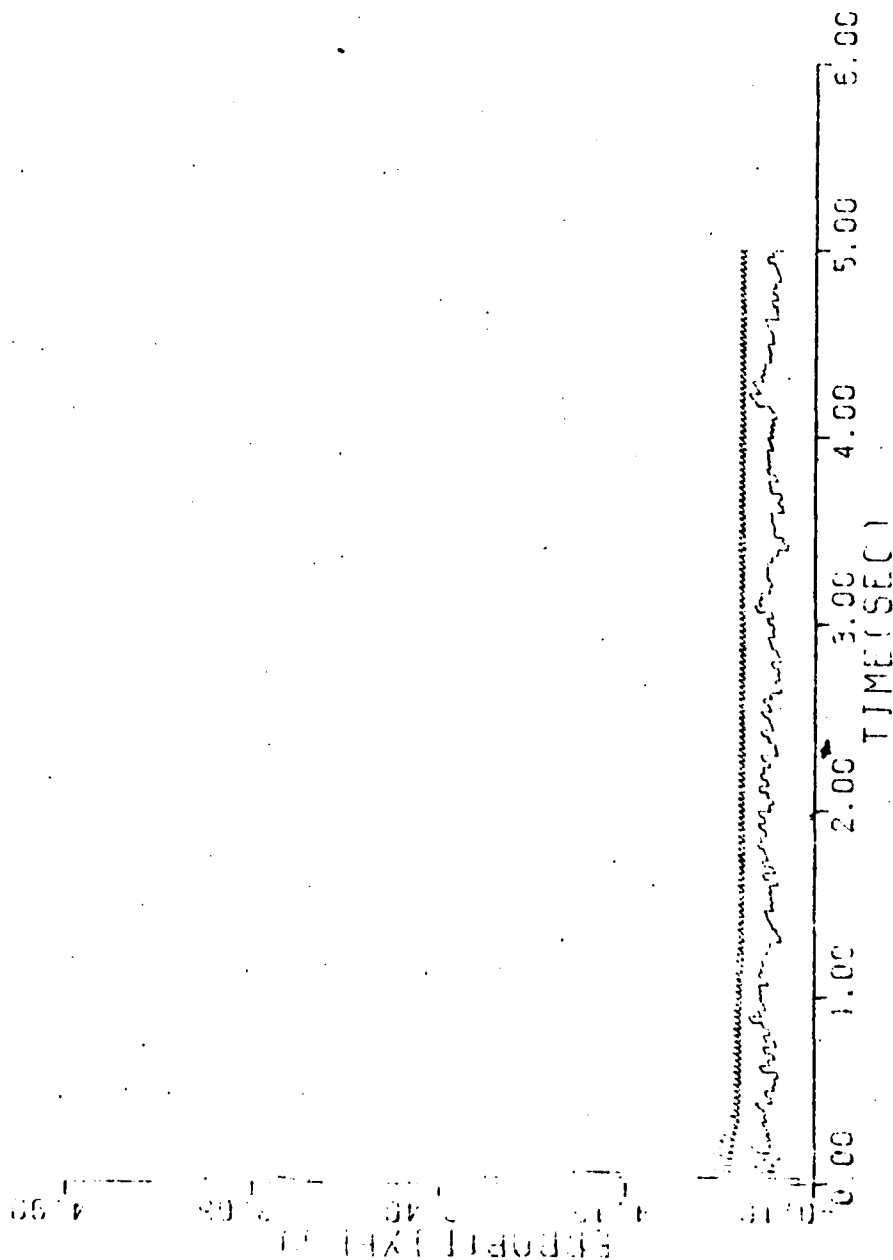


Figure 38b. Case 1

ESTIMATED X-MINUS POSITION (+/-) SIGMA

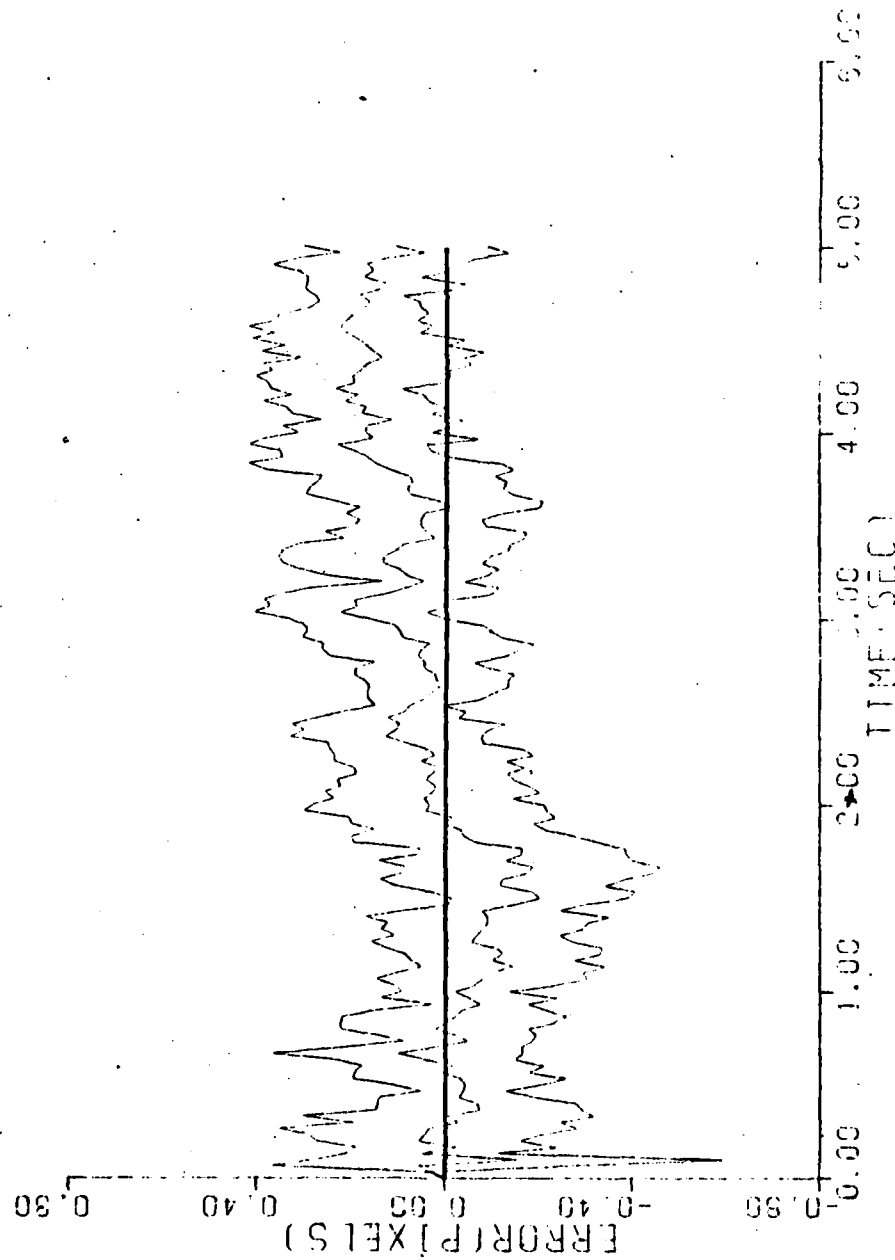


Figure 38c. Case 1

ESTIMATED Y-MINUS POSITION (+/-) SIGMA

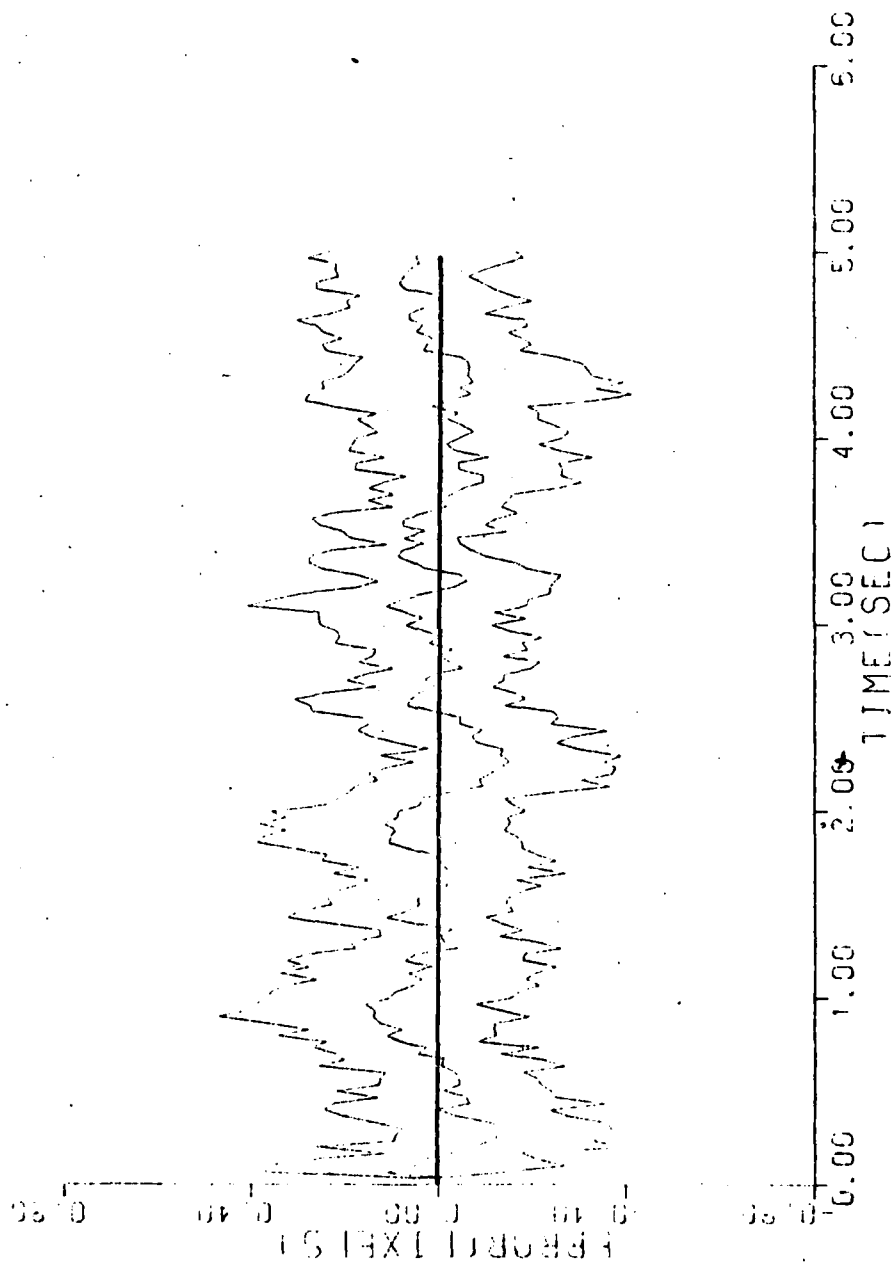


Figure 38d. Case 1

ESTIMATED /-PLUS POSITION (1+/-) SIGMA

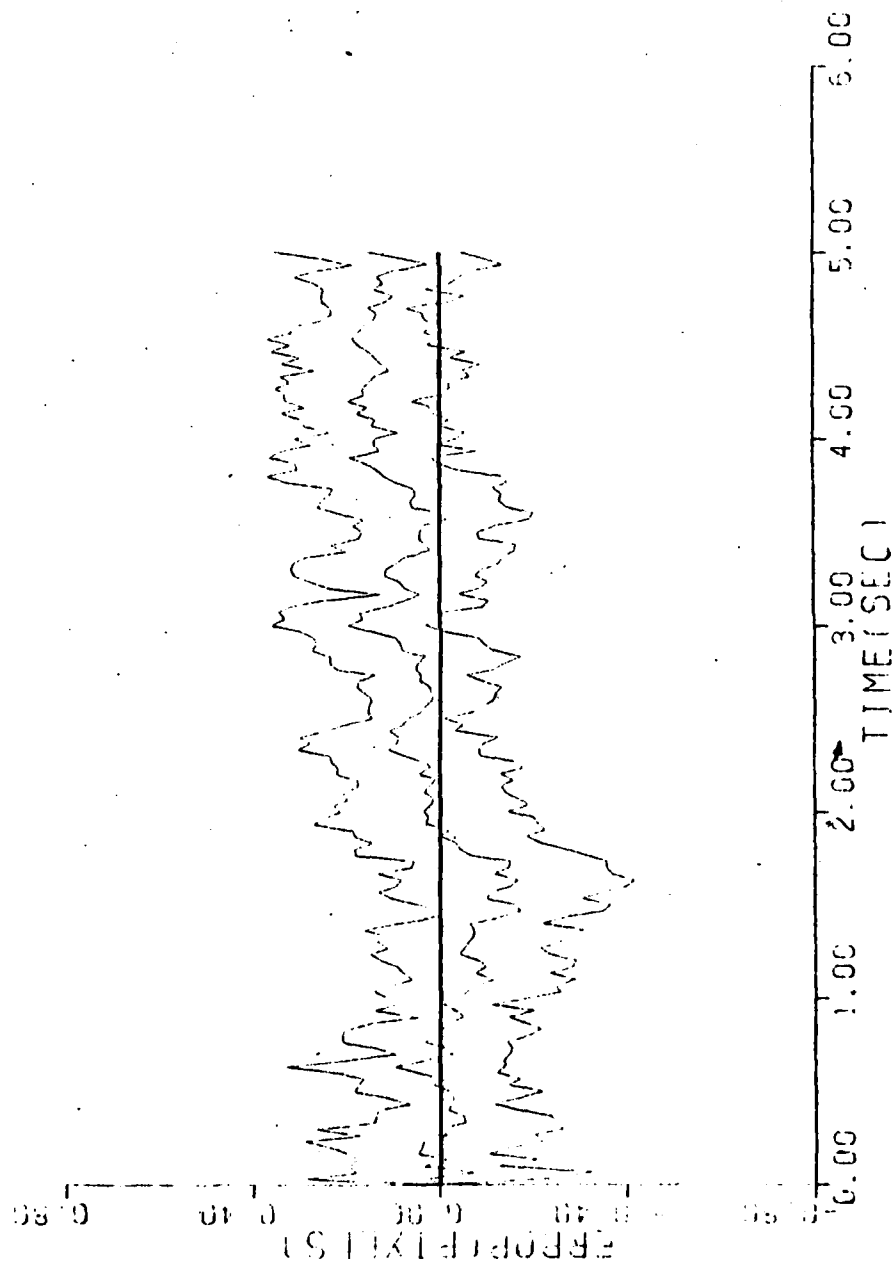


Figure 38e. Case 1

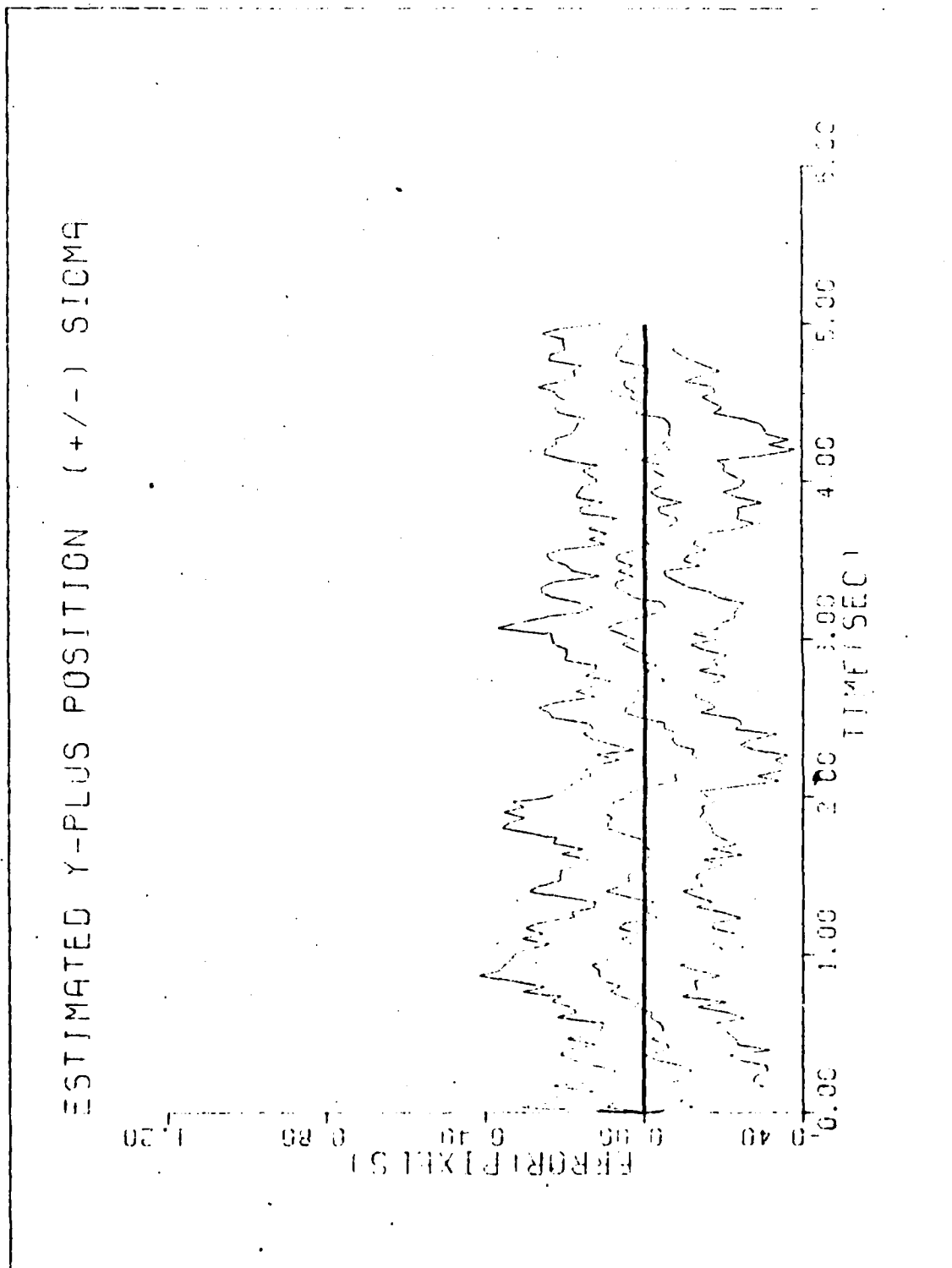


Figure 38f. Case 1

ESTIMATED λ -MINUS CENTROID POSITION (+/-) SIGMA

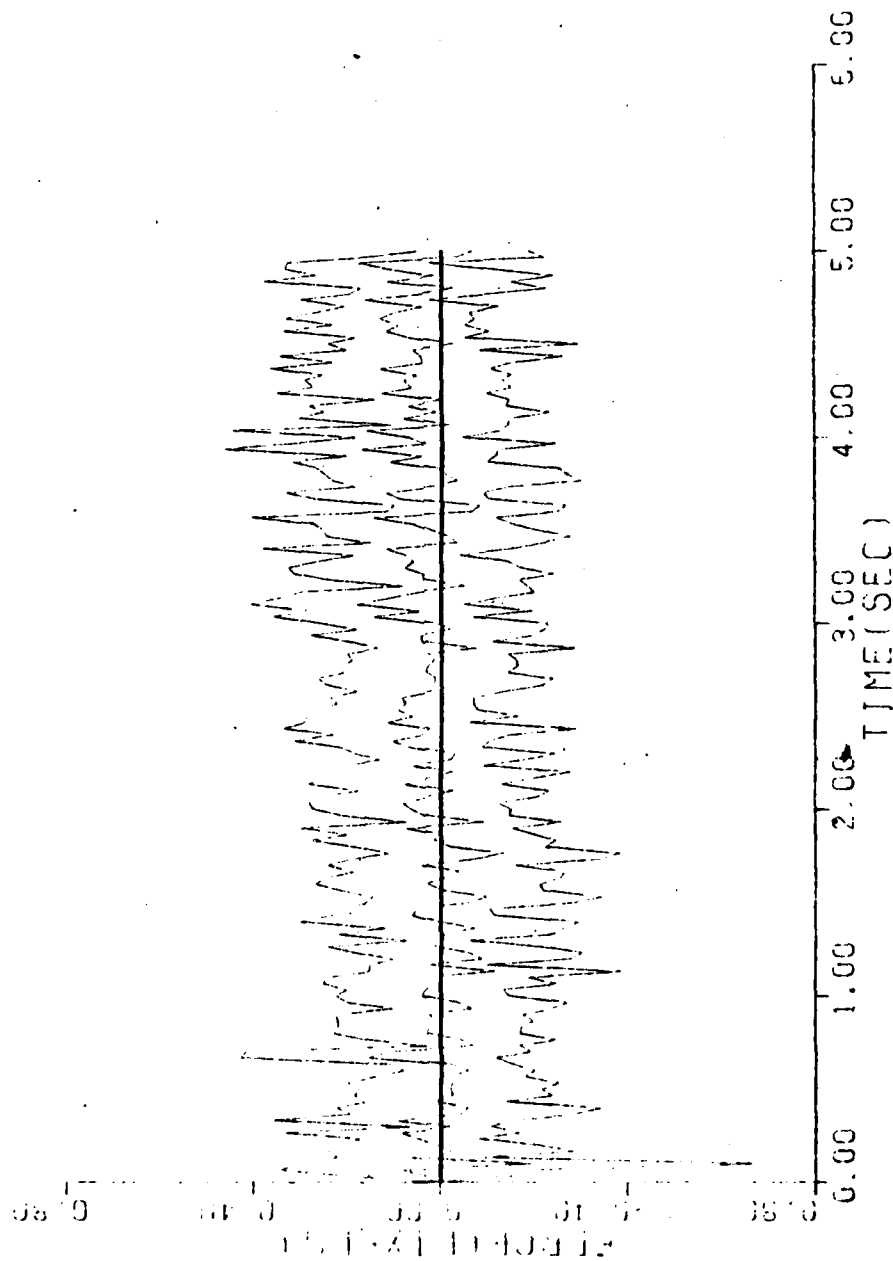


Figure 38g. Case 1

ESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA

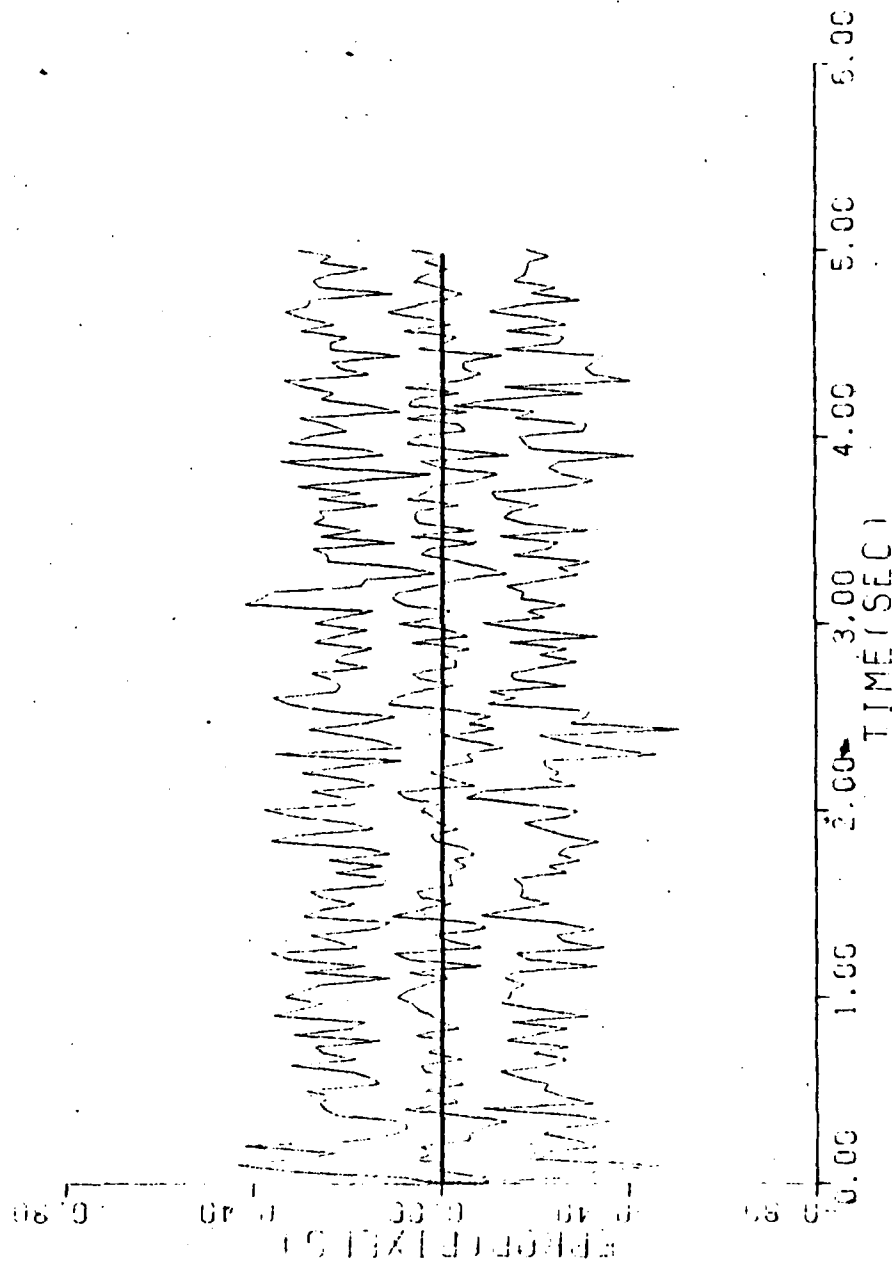


Figure 38h. Case 1

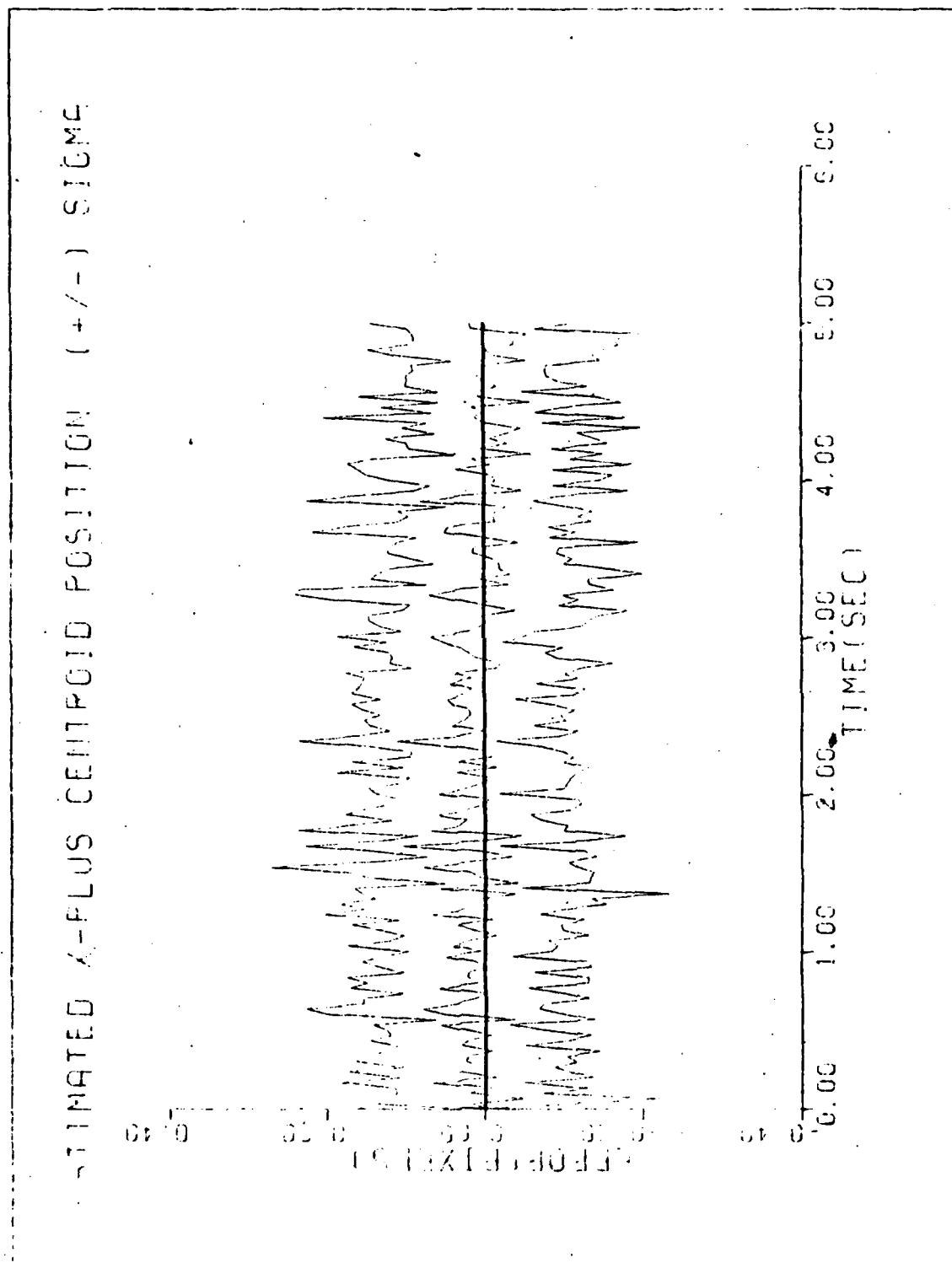


Figure 38i. Case 1

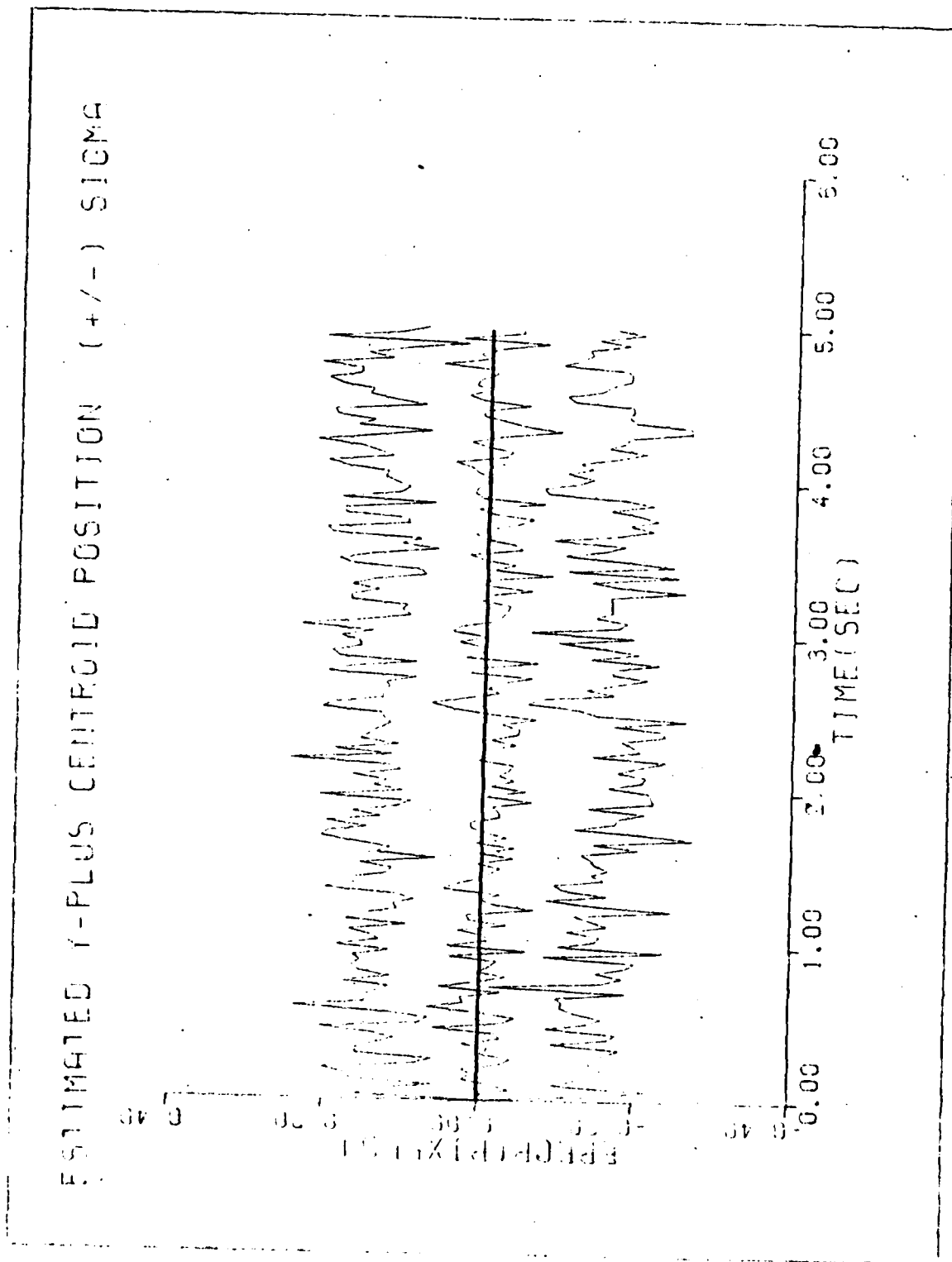


Figure 38j. Case 1

i.e., it was "conservatively" tuned. The reason for this conservative tuning which is used to guard against possible filter divergence is explained by referring to Figure 35c. As shown, the filter mean error has a transient characteristic which lasts for approximately two seconds. In an attempt to minimize this transient, the conservative tuning was used. The reason for this unexpected transient characteristic is not completely understood. However, a discussion of possible causes is included in the next chapter. Also note the filter was given perfect knowledge of the initial conditions through $\hat{x}_f(t_0^+)$.

5.7 Analysis of Filter Performance

The results of the filter's performance will be presented in tabular form in this chapter and in graphical form in Appendix C. The error statistics shown in the tables are the temporal averages of the mean errors and standard deviations from $t = 3.5$ to $t = 5.0$. For the truth model parameters, the following standard values were used unless specifically noted in the comments, section of the table:

- a) Aspect Ratio (AR) = 1
- b) Intensity Function Spread Parameter (σ_{pv}^2) = 2
- c) Signal to Noise Ratio (SNR) = 20
- d) Maximum Hot Spot Intensity (I_{max}) = 20

Thus, combining c and d above yields a rms background noise value = 1.

The first line in the header entry for each of the tables gives those parameters which were used for filter tuning where τ_{df} and σ_{df}^2 are as previously defined and α is the exponential smoothing parameter shown in Equation (1-4). The truth model parameters which were often varied from one Monte Carlo run to another are given in the second

line of the header column. The first entry references the trajectories defined in section 2.3. For trajectory 2 which is the pullup maneuver, the dynamics of the turn (g-factor) are shown in the comments column. The second entry gives the roll rate, ω , for the run in rad/sec. The final truth model entry in the header column gives the number of hot spots (NUMHS) used for the run. The statistics presented in the tables are defined as:

\bar{x}_e = average of the mean error for the true position in the x-direction from $t = 3.5$ to $t = 5.0$ at times minus and plus (similarly for \bar{y}_e)

$\overline{\sigma x}_e$ = average of the standard deviation of the true position in the x-direction from $t = 3.5$ to $t = 5.0$ at times minus and plus (similarly for $\overline{\sigma y}_e$)

\overline{cx}_e and $\overline{\sigma cx}_e$ = errors as defined above for the centroid position (similarly for \overline{cy}_e and $\overline{\sigma cy}_e$)

5.8 Evaluation of Correlation Methods

Based on the analysis of the correlation methods detailed in Chapter 4, the FFT and phase correlation methods were incorporated into the tracking algorithm, to evaluate the performance of each in a tracking environment. The correlators were evaluated under a benign tracking environment, trajectory 1, against targets exhibiting both single and multiple hot spots. The results of the performance evaluation are shown in Table III. For the single hot spot case, the FFT method has a smaller mean error the the y-position, while the phase correlation method produces better mean estimates in the x-direction.

Table III. Correlator Performance Results

$(\tau_{df}, \sigma^2_{df}, \alpha)$

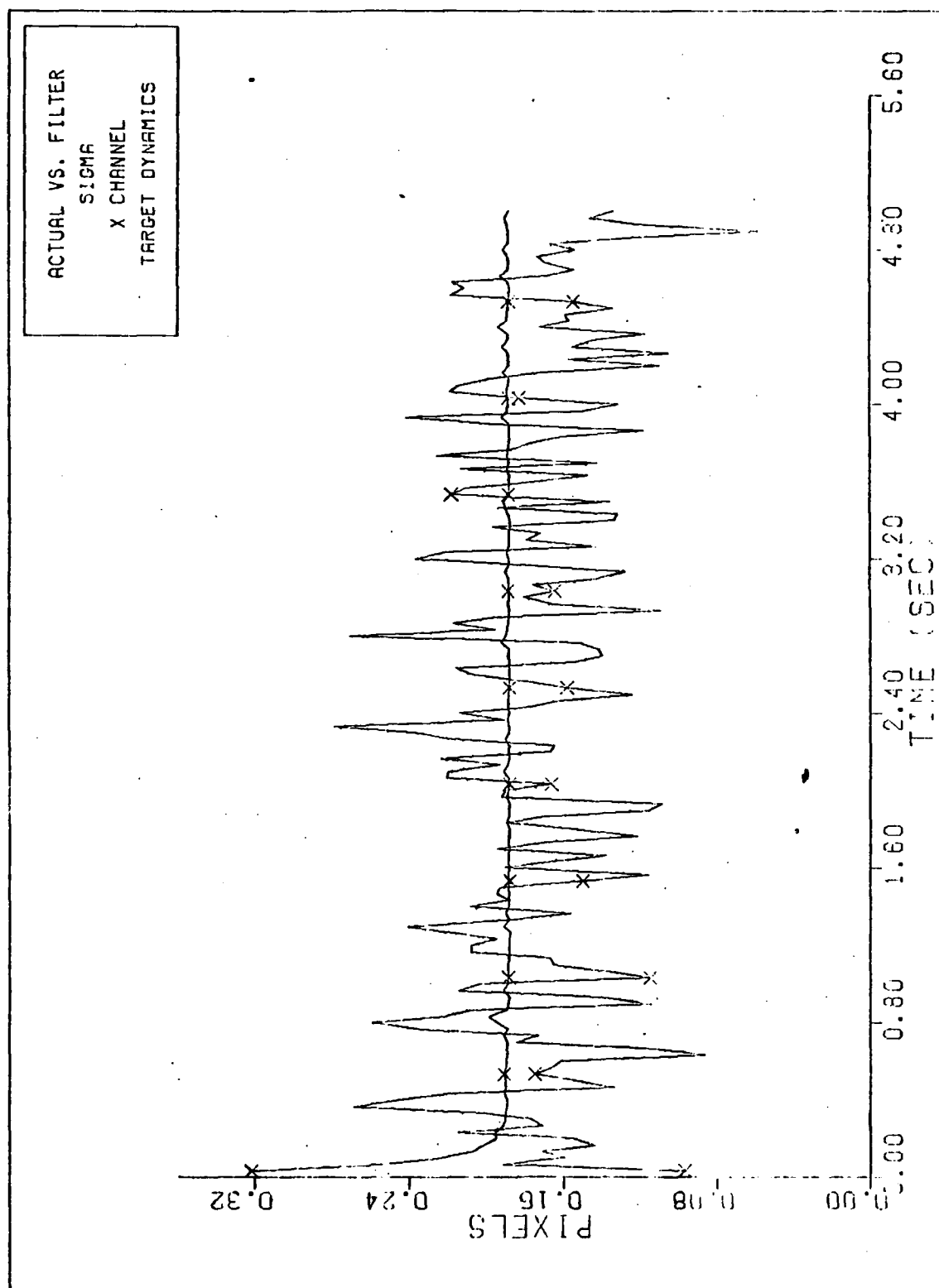
Header = (Trajectory, Roll Rate, NUMHS)

Comments	Header	$\bar{x}_e(-)/\sigma x_e(-)$	$\bar{x}_e(+)/\sigma x_e(+)$	$\overline{cx}_e(-)/\sigma cx_e(-)$	$\overline{cx}_e(+)/\sigma cx_e(+)$	$\bar{y}_e(-)/\sigma y_e(-)$	$\bar{y}_e(+)/\sigma y_e(+)$	$\overline{cy}_e(-)/\sigma cy_e(-)$	$\overline{cy}_e(+)/\sigma cy_e(+)$
Case 1 FFT Cor	3.5,150,.05 1,0,1	.142/ .177	.104/ .162	.053/ .207	-.005/ .116	-.006/ .203	-.010/ .190	.000/ .222	-.009/ .154
Case 2 Phase Cor	3.5,150,.05 1,0,1	-.043/ .344	-.070/ .331	-.134/ .405	-.190/ .388	.112/ .326	.1081 .317	.118/ .367	.108/ .352
Case 3 FFT Cor	3.5,150,.05 1,0,3	.142/ .160	.113/ .148	.052/ .179	-.007/ .056	.010/ .170	.005/ .156	.015/ .178	.006/ .066
Case 4 Phase Cor	3.5,150,.05 1,0,3	.286/ .315	.258/ .302	.198/ .339	.139/ .292	.063/ .292	.058/ .281	.068/ .331	.059/ .292

However, the FFT method produces the best position estimates in both directions in the three hot spot cases. The data presented in Table III can be used to calculate the rms error, defined as $\text{rms} = \sqrt{m^2 + \sigma^2}$. A calculation of this statistic revealed that in all cases the phase correlation method had a rms error which was approximately 50% greater than the FFT rms error. Additionally, the standard deviation of the FFT method is substantially smaller in all cases, as expected based on the analysis of the correlation methods. Therefore, the FFT method was chosen as the correlation method to be used in the remainder of the evaluations. Since the mean tracking errors were slightly larger in the three hot spot case for the FFT method, the remainder of the evaluations were made for three hot spot targets, with case 1 to serve as the baseline for the correlator/Kalman filter performance.

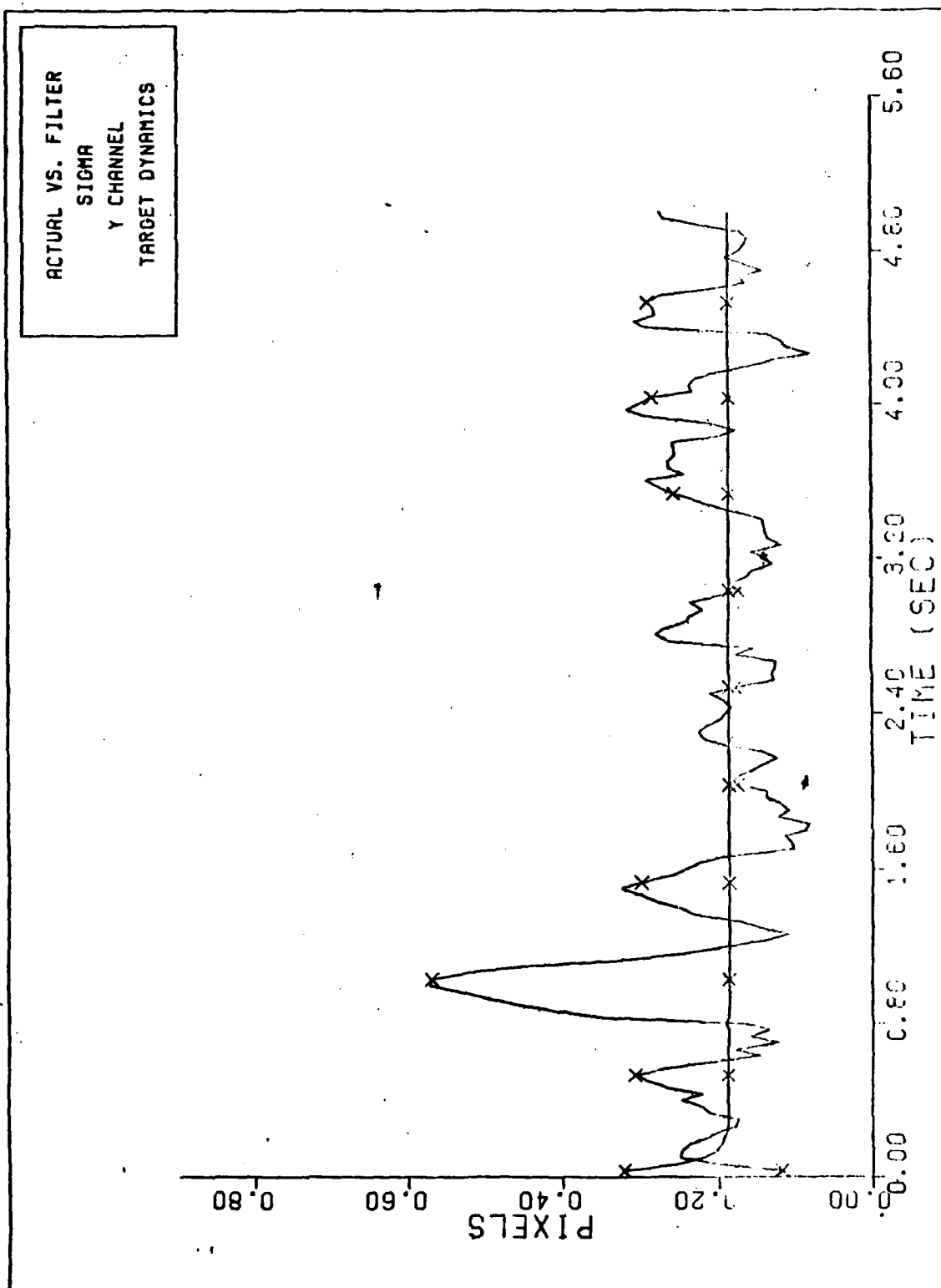
5.9 Evaluation of Harnly and Jensen EKF

With case 1 of Table III established as the performance baseline for the correlator/Kalman filter in the single hot spot case, the 8-state EKF designed by Harnly and Jensen (6), which uses a Brownian motion acceleration model and a bivariate Gaussian measurement model, was evaluated to provide a performance benchmark. The Harnly-Jensen cases are referred to by letters whereas the correlator/Kalman filter cases are numbered. The first case, case A, was conducted with the same truth model parameters used in case 1. As shown in Figures 39a and 39b, the filter is well-tuned for this scenario. Although not accomplished, extended Kalman filters are often tuned by comparing the filter computed covariance to the actual rms error instead of the actual standard deviation to minimize any bias effects. As in the previous cases, the filter was initialized with perfect state knowledge so no recovery

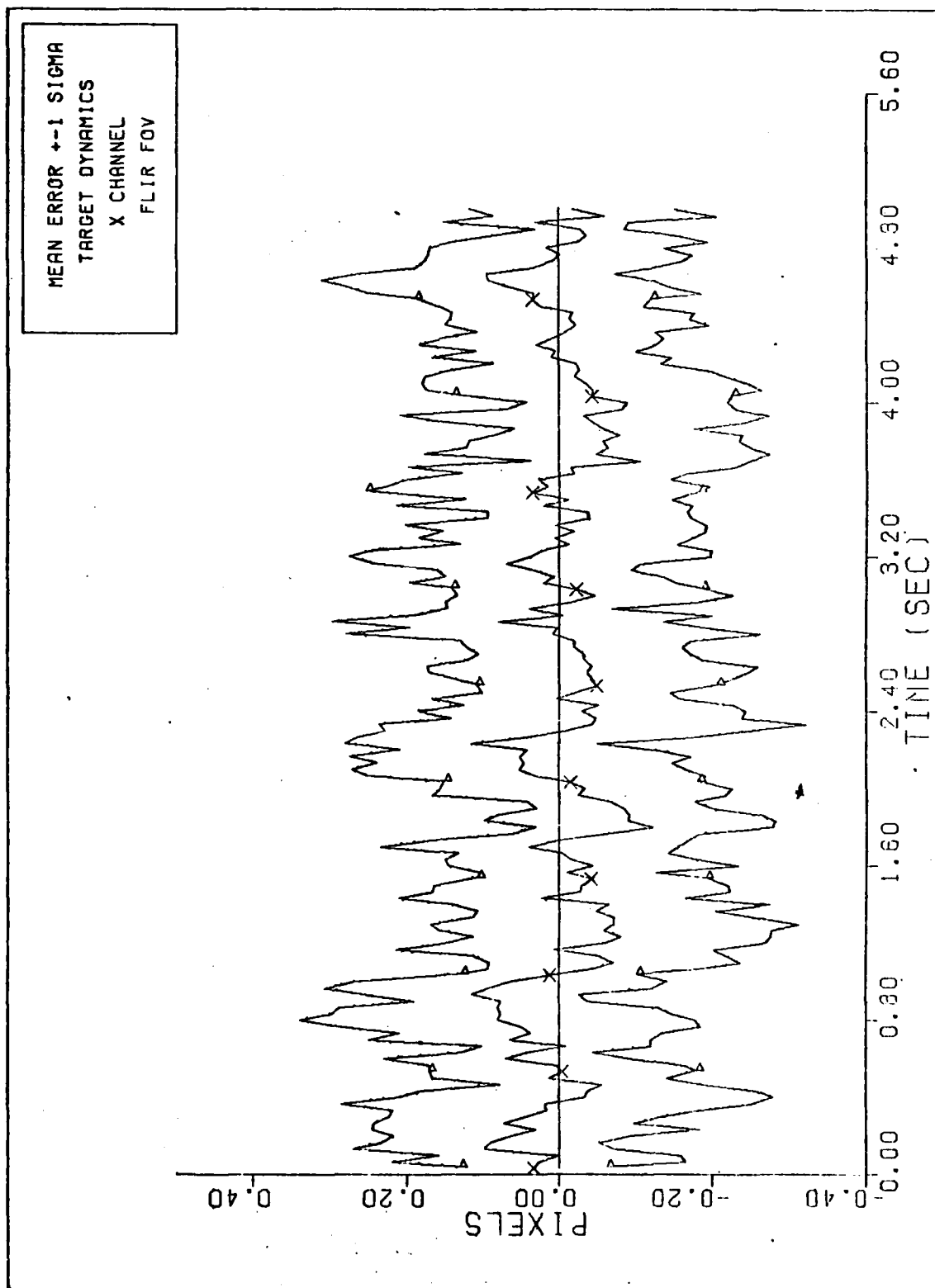


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure 39a. Case A

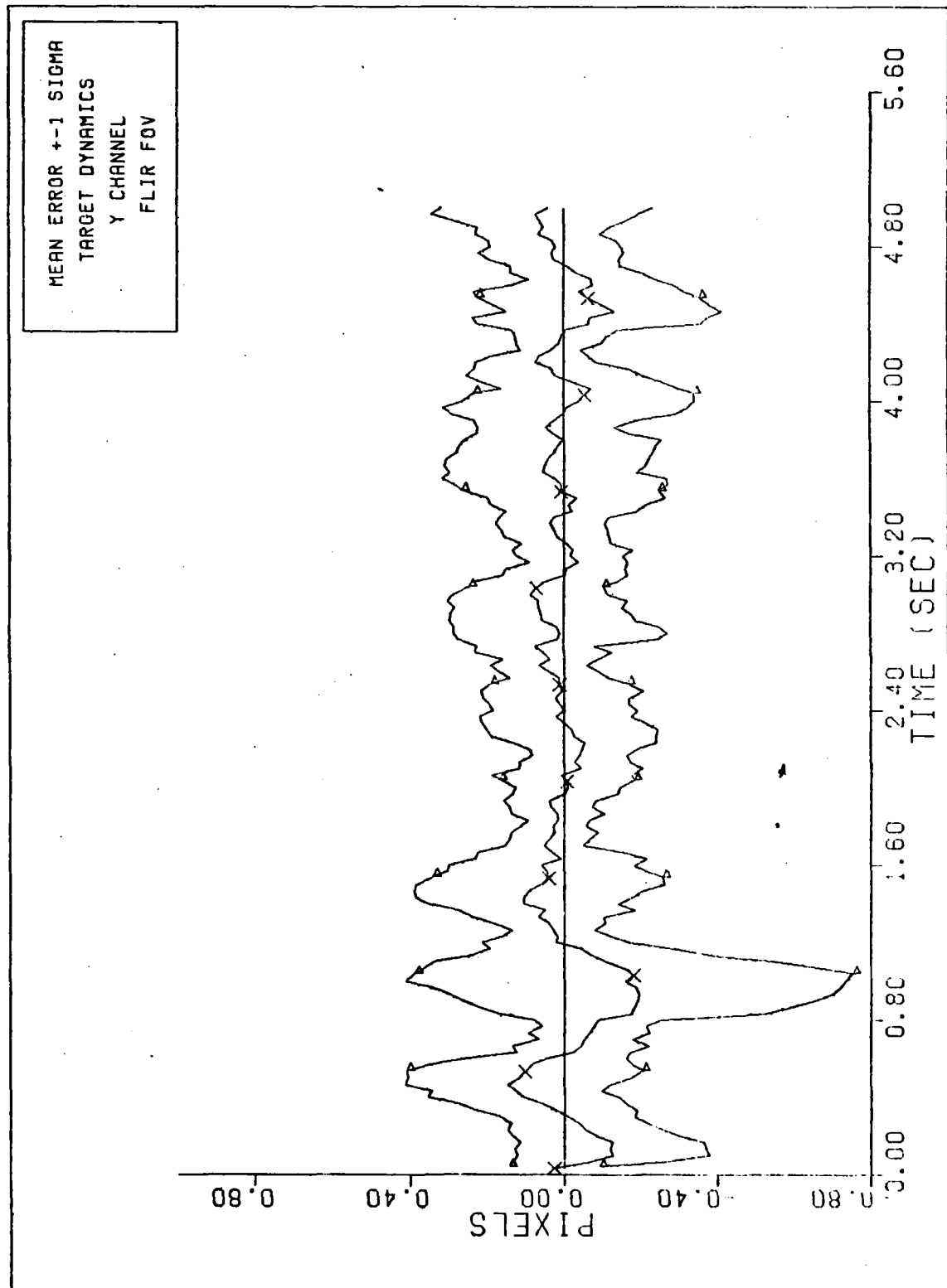


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure 39b. Case A



X CHANNEL DYNAMICS ERROR (S/N=)

Figure 39c. Case A



Y CHANNEL DYNAMICS ERROR (S/N 20)

Figure 39d. Case A

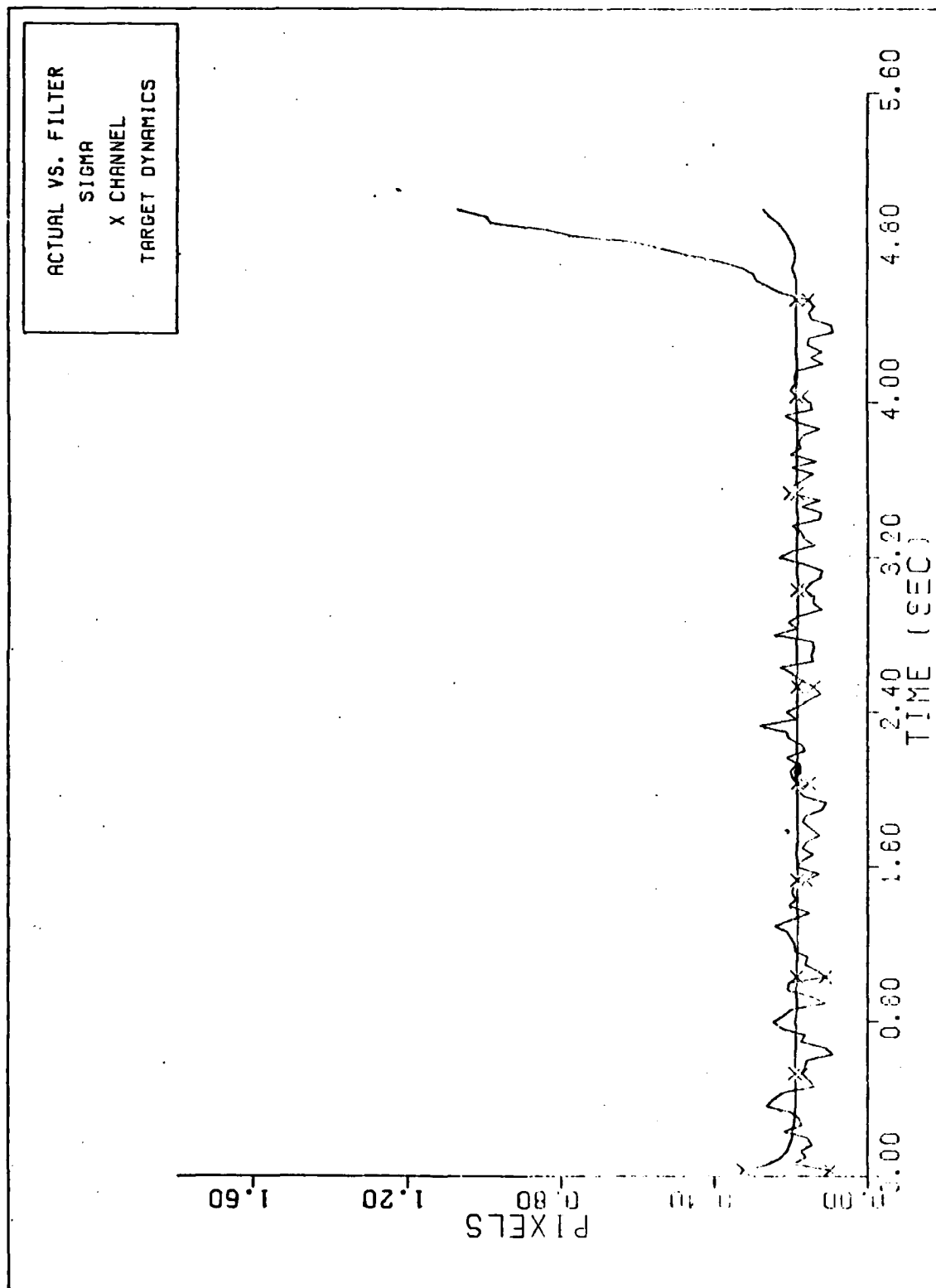
period is observed. (Note: The position plots are for the errors at time plus.) The statistics for the run were averaged as before from $t = 3.5$ to 5.0 and are shown in Table IV. (Note: The statistics shown are the only position calculations available from the Harnly-Jensen routine.) When this case is compared to case 1, it is seen that in the x-direction the mean error bias present in the correlator/Kalman filter is not present in the EKF, with which excellent tracking is obtained. However, the EKF has a slightly larger standard deviation. In the y-direction similar error characteristics are noted with the EKF having the smaller mean error and the correlator/Kalman filter having the smaller standard deviation. Calculation of the rms errors from the table data shows the rms error for the Harnly-Jensen EKF is .06 pixel less in the x-direction and .01 pixel greater in the y-direction.

Table IV. Harnly-Jensen EKF Performance
Header (σ_{df}^2)
(Trajectory)

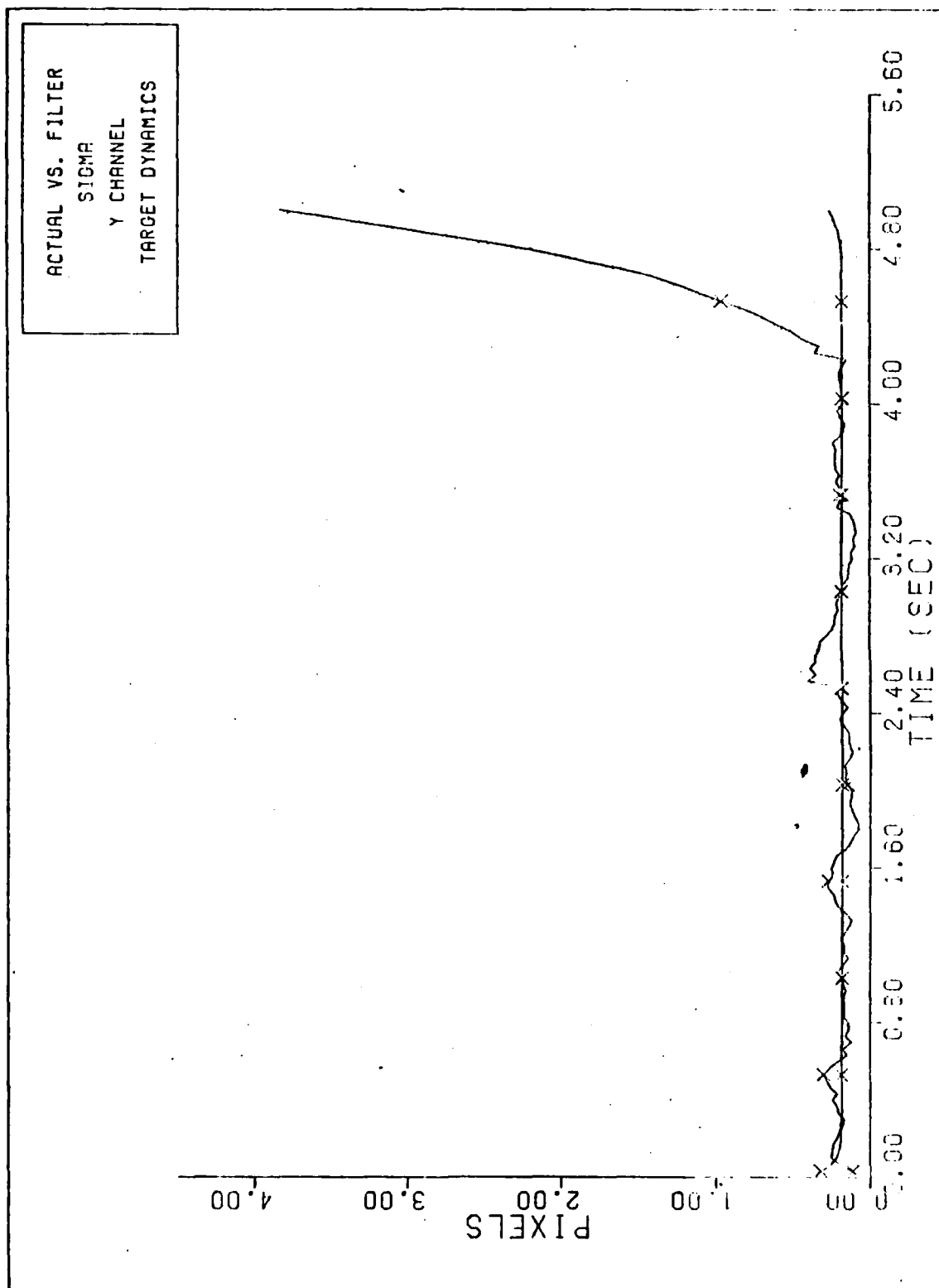
Comments	Header	$\bar{x}_{ERR}(+)/\sigma_{x_{ERR}}(+)$	$\bar{y}_{ERR}(+)/\sigma_{y_{ERR}}(+)$	Remarks
Case A	150 1	-.012/ .165	.006/ .221	
Case C	200	-.000/ .164	.015/ .184	Special Trajectory

Note: Case B statistics not given due to filter divergence at the end of simulation.

The next run with the filter, case B, was a 2-g pullup maneuver using trajectory 2. The performance plots for this case are shown in Figures 40a-d. When this case is compared to cases 12 and 13 for the correlator/Kalman filter, which are presented in section 5.12, it is seen that the EKF responds better at maneuver initiation with the mean tracking error being 0.5 pixel at the maximum point as compared to a 1.0 mean pixel error for the correlator/Kalman filter (see Figures 40d and C-12d of Appendix C). Additionally, the EKF recovers to essentially a zero mean error in 0.75 sec whereas the correlator/Kalman filter has a transient of 1.0 sec. Note in the Harnly-Jensen cases the mean errors are calculated by subtracting the filter estimated states from the true states, opposite of Equation (5-1), which accounts for the plot differences. However, at the end of this simulation the Harnly-Jensen filter estimates in the y-direction begin to diverge. At this point, the target is approaching the position of closest approach, where it will transition from closing on the tracker location to receding from the tracker location. Therefore, the FLIR frame and its reference system must rotate at a greater rate than at any other point in the simulation to keep the target centered in the FOV. This rotation creates a non-inertial acceleration, and appeared in a 6-state filter which did not model acceleration employed by Harnly-Jensen in a manner very similar to case B. However, when Harnly-Jensen added the two acceleration states to the EKF, this divergent characteristic was not observed. Since many of the truth model parameters are not the same as in the Harnly-Jensen simulations a direct comparison cannot be made. It is also noted that even at the end of case A, the filter's standard deviation in the y-direction becomes larger which may have been caused by the same source.

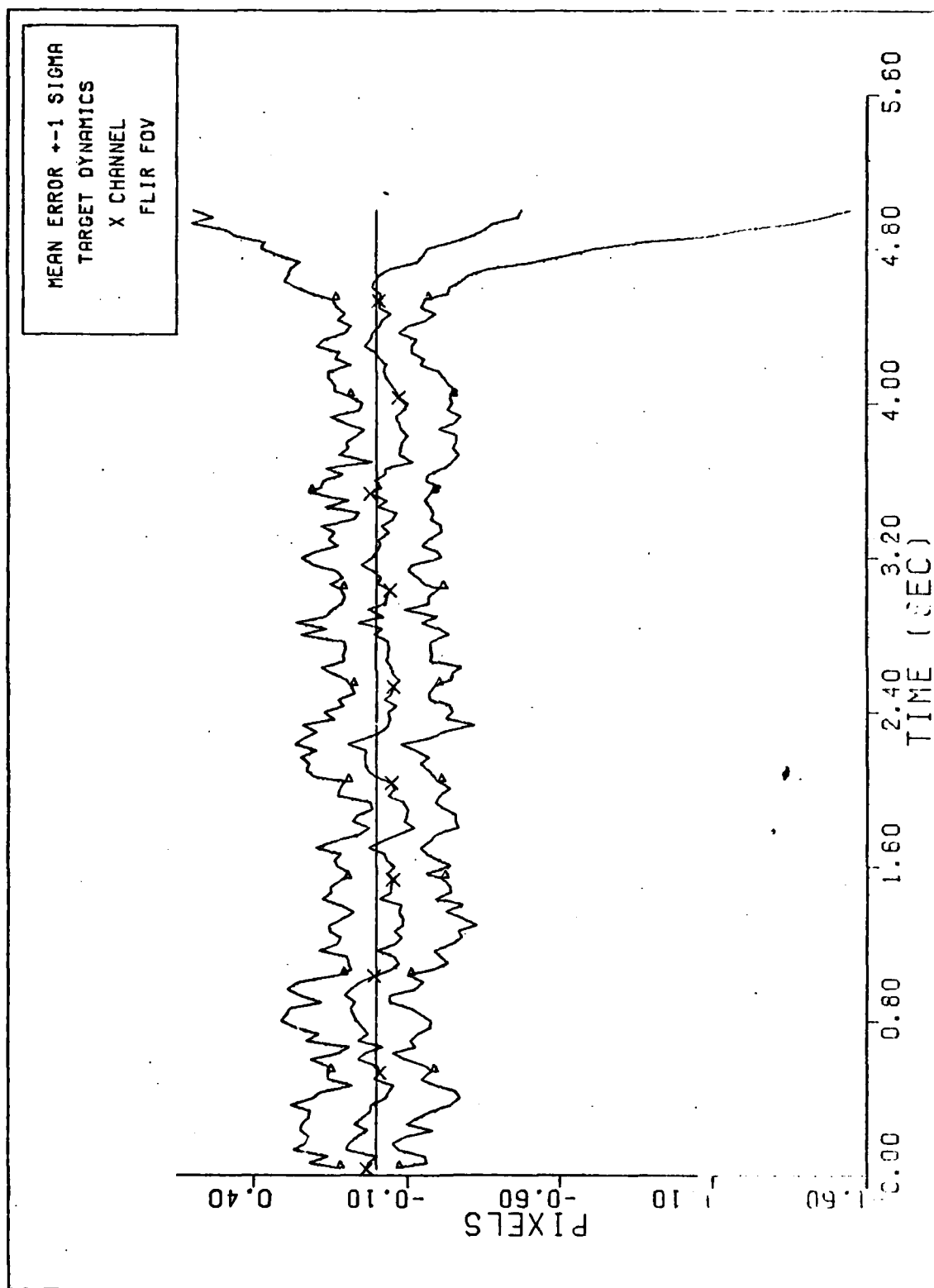


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure 40a. Case B

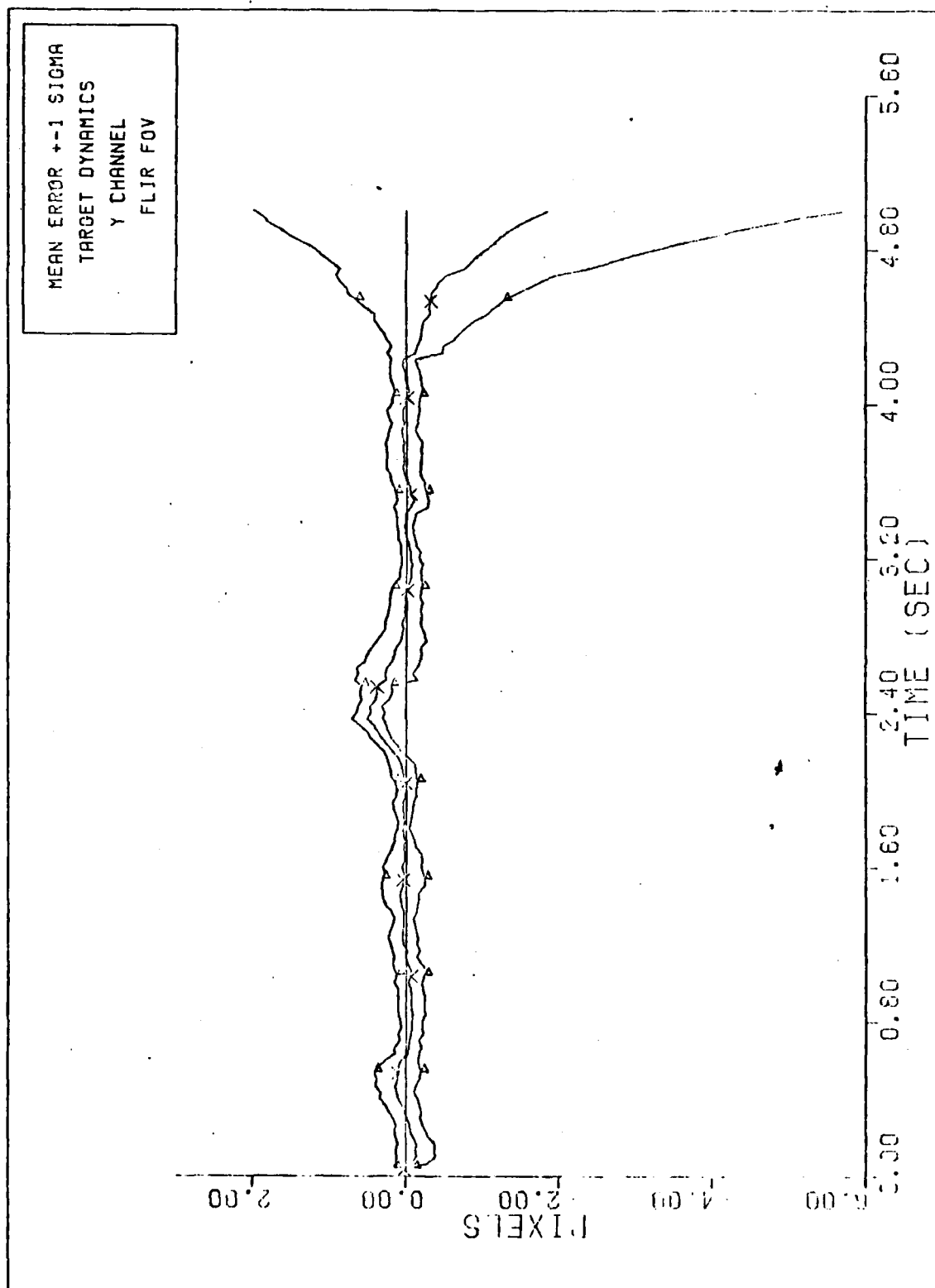


FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)

Figure 40b. Case B



X CHANNEL DYNAMICS ERROR (S/H - 20)
Figure 40c. Case B



Y CHANNEL DYNAMICS ERROR (PIXELS) 20

Figure 40d. Case B

In order to determine if the geometry of trajectories 1 and 2 had a bearing on the increase in y-position errors, a special trajectory was designed which avoided the crossover situation. For this special trajectory, the simulation was initiated with inertial coordinates

$$x_I(t_0) = 20000. \text{ m}$$

$$y_I(t_0) = 10000. \text{ m}$$

$$z_I(t_0) = 30000. \text{ m}$$

and the constant velocity used during the entire simulation was

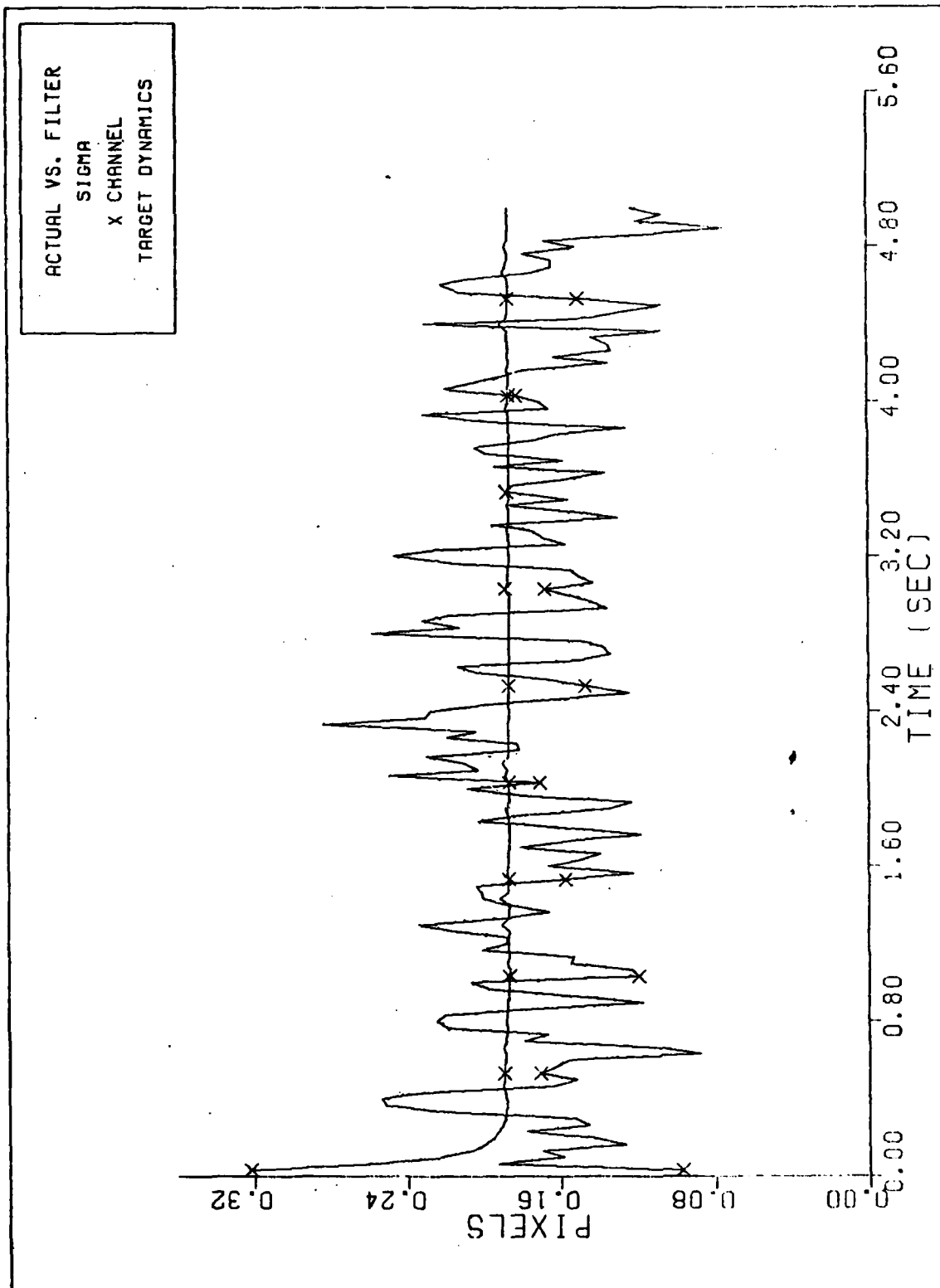
$$\dot{x}_I(t) = -500. \text{ m/sec}$$

$$\dot{y}_I(t) = -300. \text{ m/sec}$$

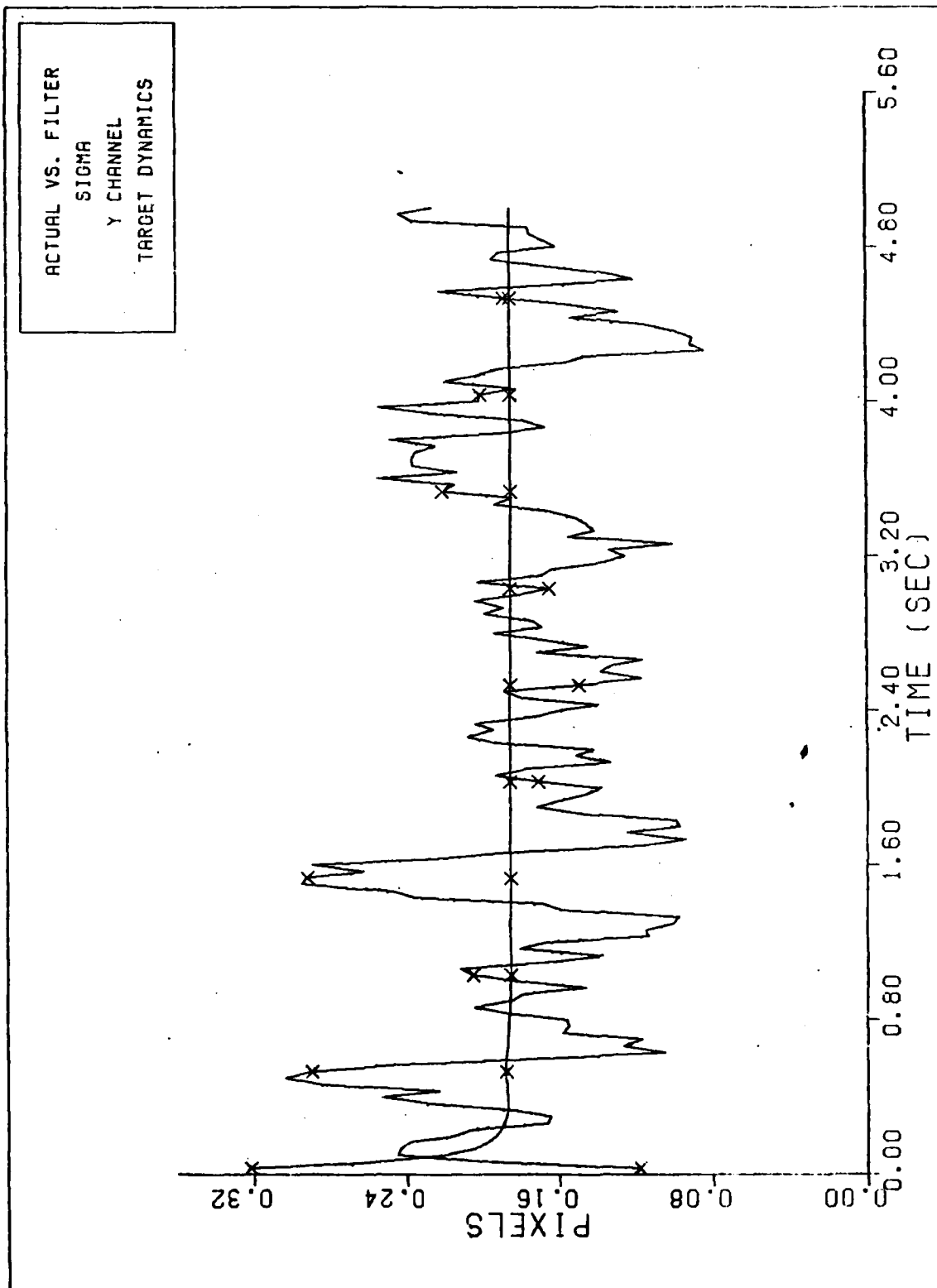
$$\dot{z}_I(t) = 0. \text{ m/sec}$$

This presents a more benign flight profile than before and the crossover point is avoided. Figures 41c and d show that in this situation the filter tracks the target very well for the entire simulation, indicating that the cause of the filter difficulties in the prior cases may be attributable to the trajectory geometry. The error statistics for this simulation are given as case C in Table IV. Also, while not identical this simulation is very similar to case 36 in the Harnly-Jensen thesis, and the observed tracking performance between these two cases are analogous. Due to the difficulties encountered with this filter, the 5-g case was not evaluated because of the similarity in the trajectory geometry. However, for future studies this case could be evaluated in a situation where the crossover geometry is avoided.

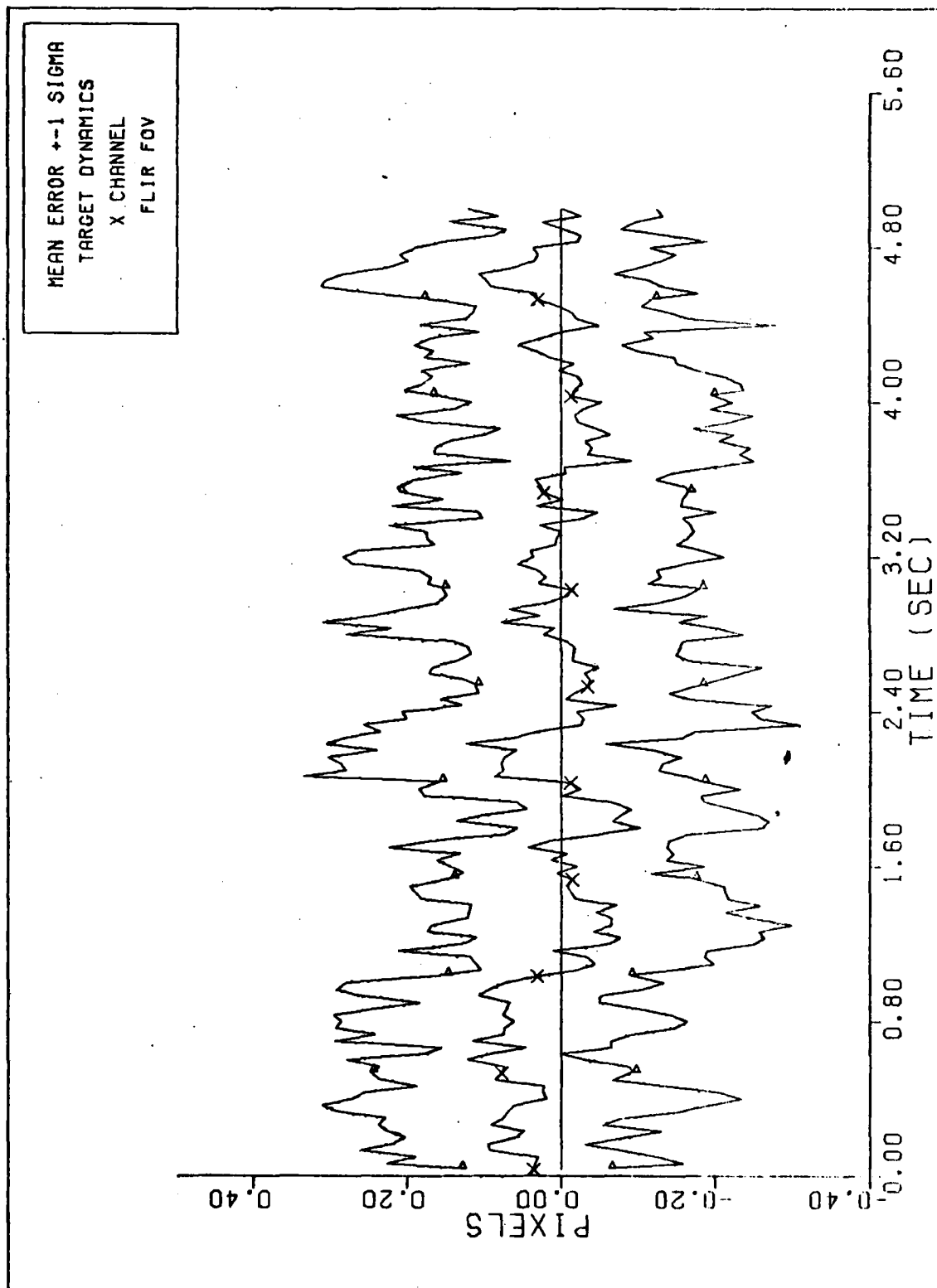
Simulations were conducted with the EKF to determine



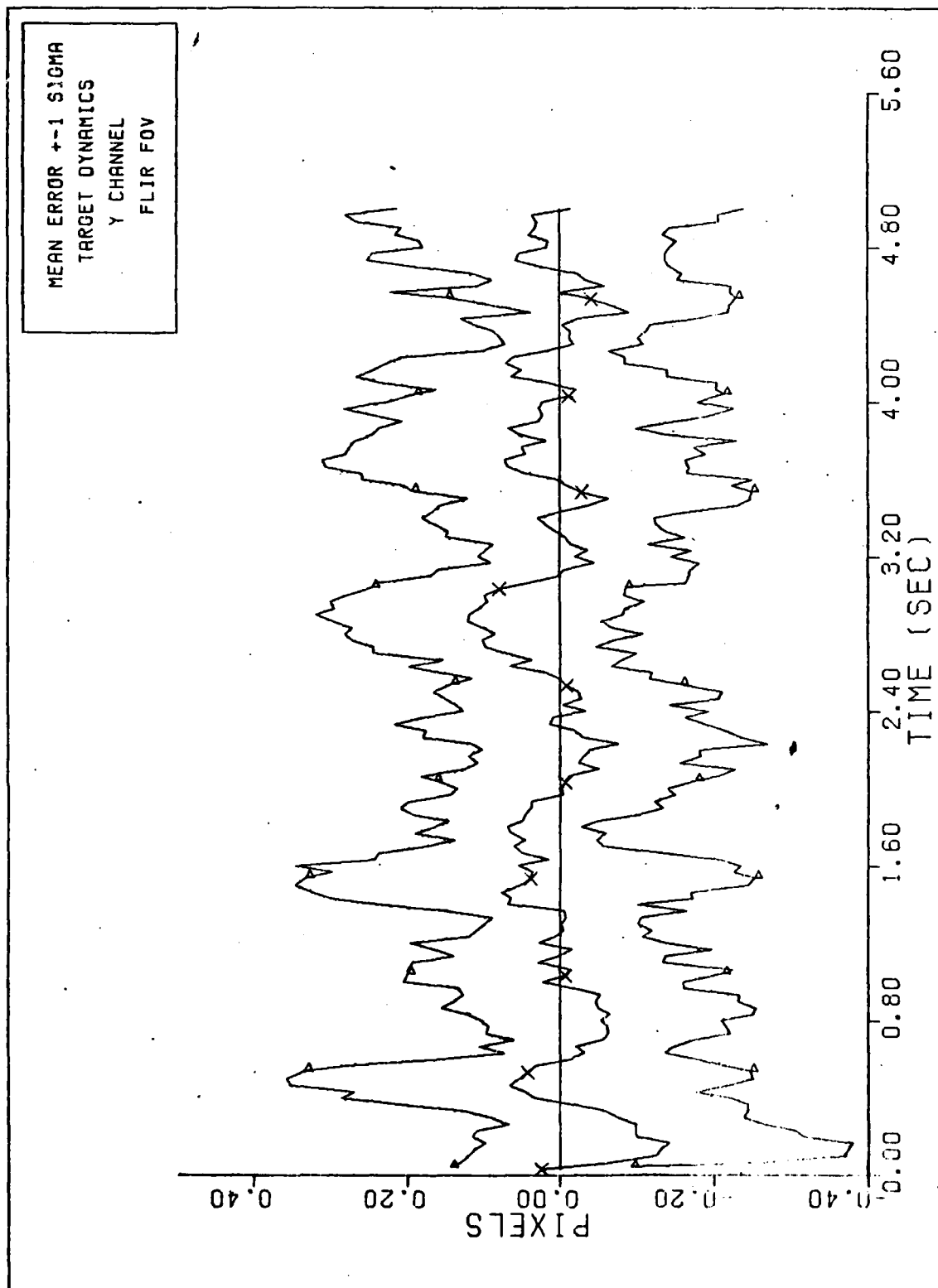
FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure 41a. Case C



FILTER VS. ACTUAL SIGMA PLOT (S/N = 20)
Figure 41b. Case C



X CHANNEL DYNAMICS ERROR (S/N=20)
Figure 4lc. Case C



Y CHANNEL DYNAMICS ERROR (S/N = 20)
Figure 4ld. Case C

how the filter would respond in a multiple hot spot environment. The hot spots were intentionally spaced so that their projection onto the FLIR image plane could be approximated as bivariate Gaussian, which is the form used in the measurement model of the EKF employed by Harnly-Jensen. To achieve this spacing, the centroid of one hot spot was initially placed at the center of the overall target centroid while the other hot spots were symmetric about the centroid at a distance of ± 1 pixel. The spread parameter, σ_g^2 , was 2. Initially, the filter tracked the target but as the simulation progresses the hot spot orientation changes. Due to this effect which results in an intensity pattern no longer well approximated as bivariate Gaussian the Monte Carlo simulations were not successful as the filter repeatedly lost target track 2 to 3 seconds into the simulations. During the phase of the simulation prior to loss of target track, the position estimates in both directions oscillated between a ± 2 pixel error which roughly corresponds to the displacement distances between the hot spot centroids on the FLIR plane. (The hot spot separation distance increases as the target approaches the FLIR location.) It appears the filter was moving from hot spot to hot spot during the estimation process.

5.10 Evaluations Using Trajectory 1.

With the baseline cases established for the one and three hot spot cases, trajectory 1 was used to evaluate how changes in α , SNR, and AR, would affect the tracking performance. The results of these runs are shown in Table V. Changing α from .05 as in case 3 to 0.1 as in case 5 results in only slight changes in the tracker's performance. In general, case 5 has slightly smaller mean errors while case 3 has slightly smaller standard deviations. The rms error is .01 pixel less for case 5 in the x-direction,

Table V. Trajectory 1 Evaluations

(τ_{df} , σ^2_{df} , α)

Header = (Trajectory, Roll Rate, NUMHS)

Comments	Header	$\bar{x}_e(-)/$ $\sigma x_e(-)$	$\bar{x}_e(+)/$ $\sigma x_e(+)$	$\bar{c}x_e(-)/$ $\sigma c x_e(-)$	$\bar{c}x_e(+)/$ $\sigma c x_e(+)$	$\bar{y}_e(-)/$ $\sigma y_e(-)$	$\bar{y}_e(+)/$ $\sigma y_e(+)$	$\bar{c}y_e(-)/$ $\sigma c y_e(-)$	$\bar{c}y_e(+)/$ $\sigma c y_e(+)$
Case 5	3.5,150,.1	.113/	.084/	.023/	-.037/	-.001/	-.005/	.004/	-.004/
	1,0,3	.168	.156	.182	.065	.174	.160	.182	.077
Case 6 SN = 10	3.5,150,.05	.134/	.105/	.044/	-.016/	.003/	-.001/	.009/	.000/
	1,0,3	.168	.155	.185	.066	.177	.163	.187	.088
Case 7 AR = 2	3.5,150,.05	.265/	.236/	.175/	.114/	.007/	.002/	.012/	.003/
	1,0,3	.146	.137	.194	.125	.172	.158	.176	.054

while in the y-direction the rms error for the two cases is approximately the same. Since increasing the α for the case does not clearly improve or degrade the performance of the tracker, it appears that for cases where the intensity pattern is not rapidly varying, an α in the range of .05 - 0.1 is appropriate. These values will be investigated later when the intensity pattern does show motion on the FLIR image plane. For case 6, the true signal to noise ratio was changed from the standard value of 20 to 10 while the filter was told the SNR was 20. The performance of the tracker is relatively unaffected by this signal-to-noise change, as was previously shown by Mercier (10). Although the case was not tested, Mercier and Harnly-Jensen showed that while going from a SNR of 20 to 10 has very little effect on the tracker's performance, going from a SNR of 10 to 1 results in a definite degradation in the filter's performance.

The last case in Table V shows the result of changing the aspect ratio from 1 to 2. For this particular scenario, this means the spread of the intensity pattern is twice as great in the x_{FLIR} direction as in the y_{FLIR} direction. As a result, the tracking ability is relatively unaffected in the y_{FLIR} direction, while in the x_{FLIR} direction an appreciable increase in the mean tracking error is noted. Thus, it appears that when the spread of the intensity function is increased, the tracker, and most probably the correlator, has difficulty determining the location of the centroid of the intensity function.

5.11 Evaluation for Rolling Maneuvers.

To evaluate the performance of the tracker when the intensity patterns were rapidly changing on the FLIR image plane and to evaluate how changing α affected the tracker's performance under this circumstance, a roll maneuver was used. Trajectory 1 was used for the target

flight path to provide a benign trajectory to minimize possible errors due to trajectory effects and to concentrate on how the changing target shape effects are handled by the tracker. Roll rates of 0.5 rad/sec and 1.0 rad/sec were used. In both instances, the roll maneuver was initiated at $t_1 = .5$ sec so the filter would have acquired the target prior to roll initiation, and the roll maneuver continued at a constant rate for the remainder of the simulation. This results in a total roll angle of 128.9° and 258.0° for the two cases respectively. The results of these runs are shown in Table VI.

A general observation on the tracking performance against the rolling maneuver is that the mean errors are slightly less in the x-direction and slightly greater in the y-direction when compared to the non-rolling cases. For the crossing trajectory with no roll, the intensity profiles are more symmetric in the y_{FLIR} direction than in the x_{FLIR} direction, and recalling the correlator performance analysis (Figure 36), the correlator performs much better in the symmetric case. This plus the fact that there is very little motion in the y-direction helps to account for the difference in the mean errors. However, in the rolling case, the intensity profiles will be more symmetric in the x-direction at times, and in the y-direction at other times, which accounts for the general trend in the mean error changes.

For the 0.5 rad/sec roll maneuver, cases 8 and 9, the $\alpha = 0.1$ has the smaller mean errors in both the x and y position estimates as expected. This is because, for a changing intensity profile, more emphasis should be placed on the newer measurements which are more representative of the current intensity pattern. In cases 10 and 11, where a 1 rad/sec roll rate was used, a greater performance enhancement was expected for $\alpha = 0.1$ than had been seen for the previous case, as the intensity pattern is varying

Table VI. Roll Maneuver Evaluations

(τ_{df} , σ^2_{df} , α)

Header = (Trajectory, Roll Rate, NUMHS)

Comments	Header	$\bar{x}_e(-)/$ $\sigma x_e(-)$	$\bar{x}_e(+)/$ $\sigma x_e(+)$	$\overline{cx}_e(-)/$ $\sigma cx_e(-)$	$\overline{cx}_e(+)/$ $\sigma cx_e(+)$	$\bar{y}_e(-)/$ $\sigma y_e(-)$	$\bar{y}_e(+)/$ $\sigma y_e(+)$	$\overline{cy}_e(-)/$ $\sigma cy_e(-)$	$\overline{cy}_e(+)/$ $\sigma cy_e(+)$
Case 8	3.5,150,.05 1,.5,3	.109/ .161	.081/ .146	.020/ .184	-.038/ .058	.019/ .173	.014/ .160	.025/ .181	.016/ .081
Case 9	3.5,150,.1 1,.5,3	.097/ .163	.069/ .145	.008/ .185	-.051/ .060	.011/ .181	.007/ .168	.017/ .185	.009/ .088
Case 10	3.5,150,.05 1,1.0,3	.137/ .156	.108/ .141	.047/ .179	-.011/ .049	.016/ .165	.012/ .152	.022/ .177	.013/ .078
Case 11	3.5,150,.1 1,1.0,3	.144/ .155	.114/ .139	.053/ .183	-.006/ .054	.009/ .172	.005/ .160	.015/ .181	.007/ .083

at a faster rate. Additionally, it was expected that because of the increased motion on the FLIR image plane, a performance degradation would be observed. However, upon analyzing the results these expectations were not met. In the x-direction, the mean errors did in fact increase slightly but the $\alpha = .05$ case shows a mean error of .01 pixel less than for the $\alpha = 0.1$ case. In the y-direction, the mean errors showed a slight decline with the $\alpha = 0.1$ case having the smallest mean errors. At this point it can be concluded that the tracker's performance may be enhanced somewhat by a judicious choice of α , with α 's in the ranges chosen being appropriate for the trajectories of concern. However, the symmetry of the intensity profiles on the FLIR image plane affects the tracker's performance more appreciably than anticipated. This problem could possibly be rectified by modifying the means of extracting the target position offsets from the correlation function and warrants further investigation (see Recommendations, section 6.3).

5.12 Evaluation for a Two-G Pullup Maneuver

The next cases were designed to evaluate the performance of the tracker when the target performs a constant 2-g pullup maneuver in the inertial y-direction (trajectory 2). Here attention is focused on how the dynamic trajectory effects are handled by the tracker, while the intensity shape function is essentially constant (it rotates about its principle axis slowly to remain aligned with the velocity vector.) The pullup maneuver was initiated at $t = 2.0$ sec and continued for the remainder of the simulation. Under this situation, the effects of varying the smoothing parameter, α , on the tracker's performance were evaluated. Additionally, to evaluate how the filter performance would respond to a relatively small deviation from that achieved on the straight crossing trajectory without retuning, the acceleration time constant, and the strength of the dynamic

Table VII. Trajectory 2 (two-g pullup) Evaluations

Header = (τ_{df} , σ^2_{df} , α)

(Trajectory, Roll Rate, NUMHS)

Comments	Header	$\bar{x}_e(-)/$ $\sigma x_e(-)$	$\bar{x}_e(+)/$ $\sigma x_e(+)$	$\overline{cx}_e(-)/$ $\sigma cx_e(-)$	$\overline{cx}_e(+)/$ $\sigma cx_e(+)$	$\bar{y}_e(-)/$ $\sigma y_e(-)$	$\bar{y}_e(+)/$ $\sigma y_e(+)$	$\overline{cy}_e(-)/$ $\sigma cy_e(-)$	$\overline{cy}_e(+)/$ $\sigma cy_e(+)$
Case 12 2-G	3.5,150,.05 2,0,3	.146/ .167	.115/ .152	.055/ .180	-.006/ .059	-.135/ .170	-.099/ .154	-.063/ .180	.010/ .062
Case 13 2-G	3.5,150,.1 2, 0, 3	.115/ .174	.084/ .160	.024/ .186	-.038/ .072	-.149/ .176	-.113/ .160	-.076/ .184	-.003/ .075

noise were left unchanged from the previous cases. Table VII shows the statistics for the 2-g maneuver cases. Once again, varying α does not significantly affect the filter's performance. While the larger α produces smaller mean errors in the x-direction, an almost equal decrease in the mean error is seen in the y-direction for the smaller α . Also, without changing the assumed target correlation time or increasing the Kalman filter gain through increasing σ_{df}^2 , the filter's estimate of the dynamic target position trails the true target position, as shown by the negative errors in the $\bar{y}_{ERR}(+)$ columns, but the filter does track the target through the maneuver. Performance enhancement could be achieved, however, by allowing τ_{df} and/or σ_{df}^2 to be adjusted, as shown in section 5.13 for the 5-g case.

5.13 Evaluation for a Five-g Pullup Maneuver

A 5-g pullup maneuver was selected to evaluate the performance of the filter in a highly dynamic tracking environment. The effects of changing the acceleration time constant from the 3.5 sec value used for the benign trajectories to a correlation time more representative of a dynamic environment, 1.5 sec, was evaluated. Additionally, the strength of the dynamic noise was increased to model higher possible rms acceleration levels and to place more emphasis (through the filter gain calculations) on the measurements and less on the dynamics model. The value for σ_{df}^2 found by this tuning process to yield the best performance was $300 \text{ pixels}^2/\text{sec}^4$. The results of the 5-g pullup maneuver cases are shown in Table VIII. For these cases, the plots of the tracker's performance are a better indicator of the response of the filter than the simple temporal averages. As shown in Figure C-14f, the filter with the 3.5 sec correlation time does not respond well at maneuver

Table VIII. Trajectory 2 (five-g pullup) Evaluations

$(\tau_{df}, \sigma^2_{df}, \alpha)$

Header =

(Trajectory, Roll Rate NUMHS)

Comments	Header	$\bar{x}_e(-) / \sigma \bar{x}_e(-)$	$\bar{x}_e(+) / \sigma \bar{x}_e(+)$	$\bar{c}x_e(-) / \sigma \bar{c}x_e(-)$	$\bar{c}x_e(+) / \sigma \bar{c}x_e(+)$	$\bar{y}_e(-) / \sigma \bar{y}_e(-)$	$\bar{y}_e(+) / \sigma \bar{y}_e(+)$	$\bar{c}y_e(-) / \sigma \bar{c}y_e(-)$	$\bar{c}y_e(+) / \sigma \bar{c}y_e(+)$
Case 14 5-G	3.5, 300, .1	.143/	.094/	.030/	-.063/	-.374/	-.287/	-.235/	-.068/
	2, 0, 3	.184	.169	.191	.077	.177	.161	.186	.080
Case 15 5-G	1.5, 300, .1	-.023/	-.055/	-.103/	-.161/	-.602/	-.441/	-.3801	-.092/
	2, 0, 3	.193	.174	.191	.070	.174	.157	.182	.078

initiation, dropping off to a -2 pixel error in 0.5 sec. Figure C-15f shows the filter with the 1.5 sec correlation time tracks the maneuver initiation much better, with the y-error only dropping to -1.5 pixels. However, as can be seen in Table VIII, since the target continues on the constant-g pullup trajectory after the maneuver is initiated, the longer correlation time is a more accurate portrayal of the target dynamics during this phase of the trajectory and yields a filter that recovers to a better estimate of the target position. Thus, a target performing jinking maneuvers in the dynamic range shown rather than a simple constant-g pullup would be better represented by the shorter correlation time.

5.14 Adaptive Q_{fd} Estimation Evaluation

At this point, the self-tuning Q_{fd} adaptation procedure was evaluated to determine if reasonable performance could be obtained using this method. It is not expected that this method will perform better than the cases in which the parameter values were tuned off-line to optimize performance for specific trajectories. However, it is hoped that adequate performance across a wider dynamic range of scenarios can be achieved. The Q_{fd} estimation procedure was initiated at $t = 0.5$ secs. This time was selected so the filter would have sufficient time to acquire the target, where the Q_{fd} value is purposefully set to a large value to prevent the filter gains from decreasing too early in the tracking phase. Thus, at $t = 0.5$ sec, the filter is in a relatively stable tracking mode, and the adaption procedure began and continued for the remainder of the simulation. The performance was evaluated against the crossing trajectory and the 2-g pullup trajectory. The results of the Q_{fd} estimation are shown in Table IX.

Table IX. Q_{df} Estimation Evaluation

$(\tau_{df}, \sigma^2_{df}, \alpha)$

Header =

(Trajectory, Roll Rate, NUMHS)

Comments	Header	$\bar{x}_e(-)/\sigma\bar{x}_e(-)$	$\bar{x}_e(+)/\sigma\bar{x}_e(+)$	$\overline{cx}_e(-)/\sigma\overline{cx}_e(-)$	$\overline{cx}_e(+)/\sigma\overline{cx}_e(+)$	$\bar{y}_e(-)/\sigma\bar{y}_e(-)$	$\bar{y}_e(+)/\sigma\bar{y}_e(+)$	$\overline{cy}_e(-)/\sigma\overline{cy}_e(-)$	$\overline{cy}_e(+)/\sigma\overline{cy}_e(+)$
Case 16	3.5,150,.1	.117/	.017/	.058/	-.052/	.018/	-.073/	.025/	-.004/
	1, 0, 3	.195	.187	.184	.061	.198	.189	.179	.068
Case 17 2-G	3.5,150,.1	.220/	.108/	.165	.042	-1.35/	-.893	-1.16/	-.613/
	2, 0, 3	.233	.227	.232	.133	.298	.291	.315	.262

When case 16 is compared to cases 3 and 5, it is seen that in the x-direction the estimation process has a substantially smaller mean error while in the y-direction comparable performance is achieved. This is not surprising since a conservative tuning was used in the other cases to reduce the transient effects in the x-direction. Case 17 illustrates the major problem with this estimation technique. While the adaptive filter will converge to a reasonable estimation of the states, under benign trajectory conditions it reduces its gains to a point where it cannot respond quickly to harsh trajectory deviations. Referring to the Filter vs. Actual Error Plot, Figure C-17b, where the major velocity change occurs, it is clearly seen that while the maneuver is initiated at $t = 2$ sec, almost .5 sec elapses before the filter begins to respond to the maneuver. However, the gains are not increased rapidly enough and track of the target is lost. Also, note in the x-direction, Figures C-17c and e, the filter overestimates the target position due to the velocity decrease in that direction; similarly in the y-direction, Figures C-17d and f, the filter underestimates the target position due to the velocity increase. These results, consistent with the results found by Harnly-Jensen (6), show that while in cases where the estimated process is slowly changing, this estimation technique may be useful, it is not robust enough for useful application in a strongly dynamic environment.

5.15 Maneuvers Out of the x-y Plane

Trajectory 4 as described in Chapter II was motivated by the desire to evaluate the performance of the filter when the target turns in toward the FLIR plane, thereby projecting three distinct and separate ellipsoids onto the FLIR image plane. Thus, substantial trajectory offsets

are combined with large target intensity shape changes, in a single and realistic scenario. However, because of the short length of the simulation, the out-of-plane angle is very small unless a very highly dynamic flight profile is used. The performance against a target flying a 2-g out-of-plane maneuver is shown in case 18. The resulting flight profile is very similar to the 2-g pullup maneuver previously discussed. Based on this, trajectory 2 was modified so that instead of initiating a 2-g pullup in the inertial y-direction, the maneuver is performed in the minus inertial z-direction. Thus, the target turns in towards the FLIR plane. Additionally, a roll rate of .25 rad/sec was used to simulate the rolling motion associated with this type of maneuver. A 10-g maneuver was used to insure the three separated target ellipsoids were projected onto the FLIR plane, yet because of the geometry of the trajectory, this is not a highly dynamic maneuver as seen in the FLIR image plane and the filter should have little difficulty tracking the target through the maneuver. The results of these two cases are presented in Table X.

For reasons previously discussed, the results of case 18 are very similar to the other 2-g trajectory cases. In case 19, the effects of the hot spot symmetry on the correlator/Kalman filter performance are seen. For this trajectory, when the target turns in towards the FLIR plane, three distinct and relatively symmetric ellipsoids are projected onto the FLIR image plane. The result can be seen in the reduced mean errors in the x-position, again emphasizing the dependence of accurate target tracking on hot spot symmetry. Also, the plots of Figure C-19a and c show that when the velocity in the x-direction is reduced, the filter slightly overestimates the movement of the target initially but quickly recovers.

The final evaluation, case 20, was made using

Table X. Out of Plane Maneuvers

(τ_{df} , σ^2_{df} , α)

Header =

(Trajectory, Roll Rate, NUMHS)

Comments	Header	$\bar{x}_e(-) / \sigma x_e(-)$	$\bar{x}_e(+) / \sigma x_e(+)$	$\bar{cx}_e(-) / \sigma cx_e(-)$	$\bar{cx}_e(+) / \sigma cx_e(+)$	$\bar{y}_e(-) / \sigma y_e(-)$	$\bar{y}_e(+) / \sigma y_e(+)$	$\bar{cy}_e(-) / \sigma cy_e(-)$	$\bar{cy}_e(+) / \sigma cy_e(+)$
Case 18	3.5,250,.1	.101/	.073/	.018/	-.035/	-.148/	-.114/	-.082/	-.015/
2-G OP	4, 0, 3	.177	.163	.186	.073	.174	.158	.180	.066
Case 19	3.5,150,.1	.045/	.051/	.017/	.029/	-.044/	-.051/	-.026/	-.020/
z-dir	2, .25, 3	.141	.128	.183	.078	.160	.148	.179	.080

trajectory 3 to test the performance of the filter when the target initiated and terminated a maneuver. A 2-g pullup was initiated at $t = 2$ sec, and at $t = 3.5$ sec the maneuver was terminated and the target continued at a constant velocity. Because of the transient over the last 1.5 seconds of this simulation, no time-averaged statistics were calculated for a tabulation. Instead, a discussion of the performance shown in the plots is given. Figures C-20c and e show very little effects of the maneuver in the x-direction, which is expected since the change in the velocity in this channel is very small. Figures C-20f and f show the filter underestimating the target position when the maneuver begins, and then when the velocity increase is terminated at $t = 3.5$, the filter overestimates the target movement. However, after a second transient period the filter recovers to good position estimates.

5.16 Summary of Test Cases

This section presents a brief recapitulation of the 20 test cases in tabular form. These cases are presented in Table XI, where deviations from the standard truth model parameters given in section 5.3 as well as the filter tuning parameters are shown. Major trends observed in these cases as well as conclusions drawn from these trends and recommendations for possible performance enhancement measures are discussed in Chapter VI.

Table XI. Summary of Test Cases

Case	Traj	Pullup Rate	Roll Rate	NUMHS	τ_{df}	σ_{df}^2	Alpha	Misc.
1	1	0	0	1	3.5	150	.05	
2	1	0	0	1	3.5	150	.05	phase corr.
3	1	0	0	3	3.5	150	.05	
4	1	0	0	3	3.5	150	.05	phase corr.
5	1	0	0	3	3.5	150	.1	
6	1	0	0	3	3.5	150	.05	SN=10
7	1	0	0	3	3.5	150	.05	AR= 2
8	1	0	.5	3	3.5	150	.05	
9	1	0	.5	3	3.5	150	.1	
10	1	0	1.0	3	3.5	150	.05	
11	1	0	1.0	3	3.5	150	.1	
12	2	2-g	0	3	3.5	150	.05	
13	2	2-g	0	3	3.5	150	.1	
14	2	5-g	0	3	3.5	300	.1	
15	2	5-g	0	3	1.5	300	.1	
16	1	0	0	3	3.5	150	.1	\hat{Q}_{fd}
17	2	2-g	0	3	3.5	150	.1	\hat{Q}_{fd}
18	4	2-g	0	3	3.5	250	.1	
19	2	10-g	0	3	3.5	150	.1	z-dir.
20	3	2-g	.25	3	3.5	150	.1	

VI. Conclusions and Recommendations

6.1 Introduction

This chapter presents the conclusions drawn from this research project and presents recommendations into other research efforts which could be pursued to enhance the performance of the correlator/Kalman filter and the realism of the truth model for performance evaluation purposes.

6.2 Conclusions

a. Data Processing Algorithm. The data processing algorithm which generated the estimated intensity function for use as the template in the correlation algorithm was shown to perform well in this research. With this method, almost identical performance was achieved in both the single and multiple hot spot cases. This is in direct contrast to the Harnly-Jensen EKF, which experienced difficulty when the target intensity function was not of the assumed bivariate Gaussian form. The extra computational loading incurred with implementing this algorithm, which makes no assumptions about the intensity function shape, is well justified. However, it is possible that the bias which the tracker exhibited could be caused by the algorithm's reconstruction of the target intensity function. A discussion of a possible research approach to determine the cause of this bias will be deferred until the recommendations section. Varying the exponential smoothing parameter, alpha, to address changing target shapes did not significantly enhance or degrade the performance of the tracker. An alpha in the range of 0.05 to 0.1 consistently resulted in acceptable performance.

b. Correlation Methods. Based on the results of the

analysis of the four correlation methods and a tracker evaluation of two methods, the FFT method clearly yielded the best tracking performance when implemented with the Kalman filter. This method performed well in both the single and multiple hot spot cases. However, the errors observed from the correlator/Kalman filter were closely tied to the symmetry of the target intensity pattern as projected onto the FLIR image plane which directly relates to the chosen method of deriving the peak offsets from the correlation function. In the asymmetric case, the center of mass calculation which was used to approximate the peak of the correlation function gives biased results which are readily apparent in the tracker's performance. Alternatives to the center of mass method will be considered in the recommendations section.

c. Kalman Filter. The first-order Gauss-Markov model chosen to represent target acceleration provided acceptable tracking performance, and thus the computational savings associated with utilizing a linear filter over an extended Kalman filter or other non-linear filters were realized. The filter proved to be robust enough to track a benign target and a five-g target, which at the ranges considered is a very dynamic maneuver, without changing the acceleration time constant and with only a slight increase in the dynamics driving noise. However, better performance was realized when the acceleration time constant selected was more representative of the maneuver being tracked. Thus, the concept of implementing a correlation algorithm in cascade with a linear Kalman filter in a dynamic environment is a viable alternative to other non-linear approaches.

d. Adaptive Q_{fd} Estimation. The adaptive Q_{fd} estimation

technique did not prove robust enough to be used throughout an entire simulation in a very dynamic environment. However, as detailed by Harnly and Jensen (6), this routine does contain indicators which could be monitored to detect when the target initiated a maneuver. Based on these indicators, the gain values within the filter could be changed to other preselected values, as an alternative to implementing more computationally burdensome maximum likelihood estimation techniques or multiple modeling schemes to provide the desired adaptation (13: Chapter 10).

e. Harnly and Jensen EKF. Assuming the divergence problem which appeared in the simulations in this research could be readily solved, the Harnly-Jensen designed EKF performed well in the single hot spot case where the target intensity function was well represented as bivariate Gaussian. However, the filter did not perform well when the target intensity pattern was not well represented as bivariate Gaussian, as expected. While this method may not readily adapt from the single to the multiple hot spot case, the performance in the single hot spot case was very similar to the correlator/Kalman filter without the increased computational loading incurred by using the data processing algorithm of this thesis. Therefore, for the single hot spot case, such as an air-to-air missile where information as to the target type could be readily obtained from the acquisition source, this may be the preferred filter.

6.3 Recommendations

a. Data Processing Algorithm. A possible cause of the bias observed in the correlator/Kalman filter is the data processing algorithm that generates the target intensity shape function. To determine if this algorithm is the cause

of the bias, a comparison between this research and the research conducted concurrently by Lieutenant Mark Kozemchak, who used the same data processing algorithm in an extended Kalman filter, should be made. If the same characteristic bias is noted, a further confirmation could be obtained by replacing the estimated target intensity function with the true target intensity function in the correlation algorithm. If the bias is noted in the Kozemchak thesis and replacing the estimated intensity function with the true intensity function removes the bias then the data processing algorithm would have to be analyzed to determine where a phase shift in the transformed domain could occur which would cause this effect. If the bias is not observed in the Kozemchak thesis, then the bias is almost certainly due to the correlation method.

b. Correlation Methods. The center of mass calculation used in this research as an approximation to the point of maximum correlation yields a bias when the intensity pattern is not symmetric. This effect noticeably degraded the performance of the tracker. An alternative formulation could be developed whereby the center of mass calculation is used to determine the approximate peak of the correlation function and then a precise peak finding routine would be employed to provide a more precise estimate. (The reason for initially using the center of mass calculation is to get in the vicinity of the peak location prior to implementing the precise peak finder. This should help avoid locking onto local peaks as a basic peak finder might.)

c. Kalman Filter. The filter showed enough robustness to track the targets of interest with only small changes to the filter parameter values. To achieve high precision performance over a wide dynamic range, this filter could be implemented in a multiple model algorithm (13:Chapter 10) in which a small number of Kalman filters

with varying parameter values would be capable of adequately modeling the dynamic profiles of concern, as an alternative to adaptive estimation of Q_{fd} in a single Kalman filter. Initial investigation of this concept conducted by Flynn (6) proved somewhat disappointing, but is possibly worth further exploration.

d. Truth Model. The truth model changes made in this research resulted in a fairly accurate portrayal of the real world processes of interest. The simulation is structured so that changes from one trajectory to another and from the single to multiple hot spot case are readily made. The truth model simulation could be enhanced by modeling the cases where the target fuselage is between the intensity function and the FLIR image plane, thereby either blocking out completely or blocking out portions of the intensity function. Realistically, when the target is receding from the tracker location the entire infrared source will be exposed to the FLIR image plane resulting in a hotter target while the converse is true for an incoming target. By varying the value of the intensity based on considerations such as these, the realism of the truth model would be improved. The unit vectors which are currently calculated for the multiple hot spot simulation could be used as the foundation upon which these more precise models could be based.

Bibliography

1. Robinson, Clarence A., Jr., "Beam-Target Interaction Testbed," Aviation Week and Space Technology, 114:14-17 (October 8, 1979).
2. Robinson, Clarence A., Jr., "Laser Technology Demonstrated," Aviation Week and Space Technology: 16-19 (February 16, 1981).
3. Singletery, James, Jr., "Adaptive Laser Pointing and Tracking Problem," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1980.
4. Westinghouse Defense and Electronics System Center. Digital Correlation Tracker, Phase I Computer Simulations, TR-74-170. Prepared for the Air Force Weapons Laboratory, Kirtland AFB, New Mexico, March 1975.
5. Maybeck, Peter S. and Daniel E. Mercier, "A Target Tracker Using Spatially Distributed Infrared Measurements," IEEE Transactions on Automatic Control, AC-25(2): 222-225 (April 1980).
6. Harnly, Douglas A. and Robert L. Jensen, "An Adaptive Distributed-Measurement Extended Kalman Filter for a Short Range Tracker," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1978.
7. Flynn, Patrick M., "Alternative Dynamics Models and Multiple Model Filtering for a Short Range Tracker," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1981.
8. Rogers, Steven K., "Enhanced Tracking of Airborne Targets Using Forward Looking Infrared Measurements," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1981.
9. Maybeck, Peter S. and Steven K. Rogers, "Adaptive Tracking of Multiple Hot-Spot Target IR Images," unpublished article, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.
10. Mercier, Daniel E. "An Extended Kalman Filter for Use in a Shared Aperture Medium Range Tracker," M.S. Thesis, Air Force Institute of Technology, Wright-Patterson AFB, Ohio, December 1978.

11. Maybeck, Peter S., et al. "Investigation of Constant Turn Rate Dynamics Models in Filters for Airborne Vehicle Tracking," unpublished article, Air Force Institute of Technology, Wright-Patterson AFB, Ohio.
12. Maybeck, Peter S., Stochastic Models, Estimation, and Control Volume I. New York: Academic Press Incorporated. 1979.
13. Maybeck, Peter S., Stochastic Models, Estimation, and Control Volume II. New York: Academic Press Incorporated. 1982.
14. Boland, J.S., et al. "Automatic Target Hand-Off Using Correlation Techniques," Auburn University, Auburn, Alabama, January 1977. (AD-A036 435).
15. Shine, James A. and Eugene A. Margerum, "Correlation of Noisy Images," Engineer Topographic Laboratories, Fort Belvoir, Virginia, September 1980. (AD-A088 659).
16. Landau, Mark I., "Radar Tracking of Airborne Targets," presented at National Aerospace and Electronics Conference, Dayton, Ohio, May 19, 1976.

Appendix A

Intensity Centroid Projection Model for Multiple Hot Spots

In this appendix, the equations required to implement the multiple hot spot projection model, described in section 2.5, are detailed.

As depicted in Figure 13, a unit vector in the $\vec{e}_{z\alpha}$ direction can be defined by

$$\vec{e}_{z\alpha} = \cos \alpha \vec{k} - \sin \alpha \vec{i} \quad (\text{A-1})$$

where

\vec{i} , \vec{j} , \vec{k} , represent unit vectors along the inertial x, y, and z, axes respectively.

To determine the direction of a unit vector in the \vec{e}_{β} direction, first rotate about the inertial y-axis.

The coordinate transformation is:

$$\begin{bmatrix} \vec{e}_{x\alpha} \\ \vec{e}_{y\alpha} \\ \vec{e}_{z\alpha} \end{bmatrix} = \begin{bmatrix} \cos \alpha & 0 & \sin \alpha \\ 0 & 1 & 0 \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix} \begin{bmatrix} \vec{i} \\ \vec{j} \\ \vec{k} \end{bmatrix} \quad (\text{A-2})$$

Now rotate about the $\vec{e}_{z\alpha}$ axis to obtain:

$$\begin{bmatrix} \vec{e}_{\beta y} \\ \vec{e}_{\beta x} \\ \vec{e}_{\beta z} \end{bmatrix} = \begin{bmatrix} \cos \beta & \sin \beta & 0 \\ -\sin \beta & \cos \beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \vec{e}_{x\alpha} \\ \vec{e}_{y\alpha} \\ \vec{e}_{z\alpha} \end{bmatrix} \quad (\text{A-3})$$

Substituting (A-2) into (A-3) yields

$$\begin{bmatrix} \vec{e}_{\beta x} \\ \vec{e}_{\beta y} \\ \vec{e}_{\beta z} \end{bmatrix} = \begin{bmatrix} \cos \beta & \sin \beta & 0 \\ -\sin \beta & \cos \beta & 0 \\ 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} \cos \alpha & 0 & \sin \alpha \\ 0 & 1 & 0 \\ -\sin \alpha & 0 & \cos \alpha \end{bmatrix}$$

$$\begin{bmatrix} \vec{i} \\ \vec{j} \\ \vec{k} \end{bmatrix} \quad (A-4)$$

Performing the indicated multiplication where $\vec{e}_{\beta y}$ in Equation (A-4) will henceforth be referred to as \vec{e}_{β} , see Figure 13, the desired vector is

$$\vec{e}_{\beta} = (\cos \alpha)(-\sin \beta)\vec{i} + \cos \beta \vec{j} + (-\sin \beta)(\sin \alpha)\vec{k} \quad (A-5)$$

The relationship between these unit vectors and FLIR plane unit vectors is

$$\vec{e}_{\beta} = \vec{e}_{y \text{ FLIR}}$$

$$\vec{e}_{z\alpha} = -\vec{e}_{x \text{ FLIR}}$$

With the $\vec{e}_{\beta}-\vec{e}_{z\alpha}$ plane translated in inertial space to coincide with the aircraft COM, Figure 14, and since the velocity vector is assumed to lie along the aircraft centerline, a unit vector out the nose of the aircraft can be determined:

$$\vec{e}_{HX} = \frac{v_x \vec{i} + v_y \vec{j} + v_z \vec{k}}{|\vec{v}|} \quad (A-6)$$

where

v_x, v_y, v_z - refer to velocity components in the inertial frame

$$\vec{e}_{HX} - \text{unit vector along aircraft velocity vector} \\ |\vec{v}| = (v_x^2 + v_y^2 + v_z^2)^{1/2}$$

Since \vec{e}_{HX} will always pass through the aircraft COM, this unit vector is used to form the first axis of a coordinate system referred to as the H frame, with its origin at the aircraft COM.

To determine the orientation of the second H-frame axis the following cross-product was used

$$\vec{v} \times \vec{j}$$

This cross-product produces a vector normal to \vec{v} and \vec{j} . Explicitly,

$$\vec{v} \times \vec{j} = \begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ v_x & v_y & v_z \\ 0 & 1 & 0 \end{vmatrix} = -v_z \vec{i} + v_x \vec{k} \quad (A-7)$$

This vector is normalized to produce a unit vector in the direction of the second H-frame axis;

$$\vec{e}_{Hy} = \frac{-v_z \vec{i} + v_x \vec{k}}{(v_x^2 + v_z^2)^{1/2}} \quad (A-8)$$

Note that this axis will always lie in the horizontal inertial plane thus the H, or horizontal, frame notation.

To find the direction of the third H-frame axis, cross the vectors which define the first two H-frame axes:

$$\begin{vmatrix} \vec{i} & \vec{j} & \vec{k} \\ v_x & v_y & v_z \\ -v_z & 0 & v_x \end{vmatrix} = v_x v_y \vec{i} - (v_x^2 + v_z^2) \vec{j} + v_y v_z \vec{k} \quad (\text{A-9})$$

Equation (A-9) is normalized to produce a unit vector in the third H-frame direction:

$$\vec{e}_{HZ} = \frac{v_x v_y \vec{i} - (v_x^2 + v_z^2) \vec{j} + v_y v_z \vec{k}}{[(v_x^2 + v_z^2)(v_x^2 + v_y^2 + v_z^2)]^{1/2}} \quad (\text{A-10})$$

Thus, a coordinate system relative to the velocity vector is defined, see Figure 15.

For trajectories where a roll about the \vec{e}_{HY} axis is desired, the two hot spots located on the \vec{e}_{HY} axis will move out of the \vec{e}_{HX} - \vec{e}_{HY} plane, whereas the hot spot 1, as defined in Figure 16, will remain along the \vec{e}_{HX} axis. To determine the location of hot spots 2 and 3, a body axis with the x-axis out the aircraft nose, the y-axis out the right wing, and the z-axis out the belly is used. As depicted in Figure 17, the direction of hot spots 2 and 3 can be determined by

$$\vec{e}_{BY} = \cos \phi \vec{e}_{HY} + \sin \phi \vec{e}_{HZ} \quad (\text{A-11})$$

where

$$\phi(t_0) = 0$$

$$\phi(t) = \omega t$$

An example is used to demonstrate how these coordinate frames are used to project the locations of the ellipsoid centers onto the FLIR image plane. Referring to Figure 18, the offset distance, δ , along the \vec{e}_{BY} axis can be projected onto the translated \vec{e}_β axis by:

$$\delta_{\beta T} = \delta [\vec{e}_{BY} \cdot \vec{e}_\beta] \quad (A-12)$$

Performing the dot product in Equation (A-12) yields

$$\begin{aligned} \delta_{\beta T} = \delta & \left[\frac{(\cos \phi)(-v_z)(\cos \alpha)(-\sin \beta) + (\cos \phi)(v_x)(-\sin \beta)(\sin \alpha)}{A} \right. \\ & + \frac{(\sin \phi)(v_x v_y)(-\sin \beta)(\cos \alpha) - (v_x^2 + v_z^2)(\sin \phi)(\cos \beta)}{B} \\ & \left. + \frac{(\sin \phi)(v_y v_z)(-\sin \beta)(\sin \alpha)}{B} \right] \quad (A-13) \end{aligned}$$

where

A = denominator of Equation (A-8)

B = denominator of Equation (A-10)

In the $\vec{e}_{z\alpha}$ direction, the projection is

$$\delta_{z\alpha T} = \delta [\vec{e}_{BY} \cdot \vec{e}_{z\alpha}] \quad (A-14)$$

or explicitly,

$$\delta_{z\alpha T} = \delta \left[\frac{(\cos \phi)(-v_z)(-\sin \alpha) + (\cos \phi)(v_x)(\cos \alpha)}{A} + \frac{(\sin \phi)(v_x v_y)(-\sin \alpha) + (\sin \phi)(v_y v_z)(\cos \alpha)}{B} \right] \quad (A-15)$$

The equations for the projection of the hot spot located along the \vec{e}_{BX} axis onto the translated $\vec{e}_\beta - \vec{e}_{z\alpha}$ plane are:

$$\delta_{\beta T} = \delta [\vec{e}_{BX} \cdot \vec{e}_\beta] = \delta \left[\frac{(v_x)(-\sin \beta)(\cos \alpha) + (v_y)(\cos \beta)}{|v|} + \frac{(v_z)(-\sin \beta)(\sin \alpha)}{|v|} \right] \quad (A-16)$$

and

$$\delta_{z\alpha T} = \delta [\vec{e}_{BX} \cdot \vec{e}_{z\alpha}] = \delta \left[\frac{(v_x)(-\sin \alpha) + (v_z)(\cos \alpha)}{|v|} \right] \quad (A-17)$$

The final step is to convert the offset distance on the translated $\vec{e}_\beta - \vec{e}_{z\alpha}$ plane into distance on the FLIR image plane. The hot spot displacements in the translated $\vec{e}_\beta - \vec{e}_{z\alpha}$ plane are normal to line of sight from the FLIR image plane and with the range known from the trajectory model, the angular displacement of the hot spots will be used to approximate linear distance on the FLIR image plane. The geometry in the \vec{e}_β direction is shown in Figure A-1.

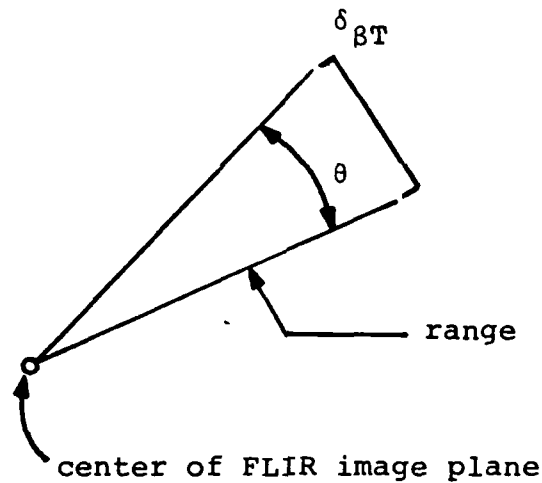


Figure A-1. Projection onto the FLIR Plane

Equation (A-18) gives the approximate displacement in the \vec{e}_{β} direction, δ_{β} , which coincides with the y_{FLIR} direction, in radians:

$$\delta_{\beta} = \tan \theta \approx \theta \approx \frac{\delta_{\beta T}}{r} \quad (A-18)$$

This distance is converted to pixels by dividing by .00002 and added to the coordinates of the aircraft COM in the FLIR plane. Thus, the offset distance in the y_{FLIR} direction for each hot spot is determined. The offset distance in the x_{FLIR} direction is determined in the same manner except the distance must be subtracted from the aircraft COM since $\vec{e}_{z\alpha} = (-)\vec{e}_{x FLIR}$.

Appendix B

Derivation of \underline{Q}_{fd} for the Kalman Filter, Chapter III

The derivation of \underline{Q}_{fd} , Equation (3-12), for the filter state equation which was briefly described in Chapter 3 will be fully detailed in this appendix. From Chapter 3, $\underline{\Phi}_f$, \underline{G}_f , and \underline{Q}_f are:

$$\underline{\Phi}_f(t_{i+1}, t_i) = \begin{bmatrix} 1 & 0 & \Delta t & 0 & A & 0 & 0 & 0 \\ 0 & 1 & 0 & \Delta t & 0 & A & 0 & 0 \\ 0 & 0 & 1 & 0 & B & 0 & 0 & 0 \\ 0 & 0 & 0 & 1 & 0 & B & 0 & 0 \\ 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{df}}} & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{df}}} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{af}}} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & e^{-\frac{\Delta t}{T_{af}}} \end{bmatrix}$$

(B-1)

where

$$A = T_{df}^2 \left[\left(\frac{1}{T_{df}} \right) (\Delta t) - 1 + e^{-\frac{\Delta t}{T_{df}}} \right]$$

$$B = T_{df} \left[1 - e^{-\frac{\Delta t}{T_{df}}} \right]$$

$$\Delta t = (t_{i+1} - t_i)$$

$$\underline{G}_f = \begin{bmatrix} 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 \\ 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 1 \end{bmatrix} \quad (B-2)$$

and

$$\underline{Q}_f = \begin{bmatrix} \frac{\sigma_{df}^2}{2T_{df}} & 0 & 0 & 0 \\ 0 & \frac{\sigma_{df}^2}{2T_{df}} & 0 & 0 \\ 0 & 0 & \frac{\sigma_{af}^2}{2T_{af}} & 0 \\ 0 & 0 & 0 & \frac{\sigma_{af}^2}{2T_{af}} \end{bmatrix} \quad (B-3)$$

where

σ_{df}^2 = assumed target acceleration process variance

σ_{af}^2 = assumed atmospheric jitter process variance

Thus, \underline{Q}_{fd} may be evaluated using (12:171)

$$\underline{Q}_{fd} = \int_{t_i}^{t_{i+1}} \underline{\phi}_{-f}(t_{i+1}, \tau) \underline{G}_{-f}(\tau) \underline{Q}_{-f}(\tau) \underline{G}_{-f}^T(\tau) \underline{\phi}_{-f}^T(t_{i+2}, \tau) d\tau \quad (B-4)$$

Performing the matrix multiplication inside the integral of Equation (B-4) yields:

$$\underline{\phi}_{-f} \underline{G}_{-f} \underline{Q}_{-f} \underline{G}_{-f}^T \underline{\phi}_{-f}^T = \begin{bmatrix} A^2 Q_{11} & 0 & ABQ_{11} & 0 & ACQ_{11} & 0 & 0 & 0 \\ 0 & A^2 Q_{22} & 0 & ABQ_{22} & 0 & ACQ_{22} & 0 & 0 \\ ABQ_{11} & 0 & B^2 Q_{11} & 0 & BCQ_{11} & 0 & 0 & 0 \\ 0 & ABQ_{22} & 0 & B^2 Q_{22} & 0 & BCQ_{22} & 0 & 0 \\ ACQ_{11} & 0 & BCQ_{11} & 0 & C^2 Q_{11} & 0 & 0 & 0 \\ 0 & ACQ_{22} & 0 & BCQ_{22} & 0 & C^2 Q_{22} & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & D^2 Q_{33} & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & D^2 Q_{44} \end{bmatrix} \quad (B-5)$$

where

A, B = are as in Equation (B-1)

$$C = e^{\frac{-\Delta t}{T_{df}}}$$

$$D = e^{\frac{-\Delta t}{T_{af}}}$$

$Q_{11}-Q_{44}$ = matrix elements of Equation (B-3)

Since the integral of the matrix shown in Equation (B-5) equals a matrix of integrals of each element, the remaining demonstration of the computation of \underline{Q}_{fd} will follow the integration of the 11-element of Equation (B-5). Substituting for A and Q_{11} yields:

$$A^2 Q_{11} = \left[(T_{df}^2) \left(\frac{\Delta t}{T_{df}} - 1 + e^{-\frac{\Delta t}{T_{df}}} \right)^2 \right] \left[2 \frac{\sigma_{df}^2}{T_{df}} \right] =$$

$$= 2T_{df}^3 \sigma_{df}^2 \left[\frac{\Delta t^2}{T_{df}^2} - \frac{2\Delta t}{T_{df}} + \frac{2\Delta t}{T_{df}} e^{-\frac{\Delta t}{T_{df}}} + 1 - 2e^{-\frac{\Delta t}{T_{df}}} + e^{-\frac{2\Delta t}{T_{df}}} \right]$$

Thus,

$$\int_{t_i}^{t_{i+1}} A^2(\tau) Q_{11}(\tau) d\tau = \{2T_{df}^3 \sigma_{df}^2\} \left[\int_{t_i}^{t_{i+1}} \frac{(t_{i+1} - \tau)^2}{T_{df}^2} d\tau \right.$$

$$- \int_{t_i}^{t_{i+1}} \frac{2(t_{i+1} - \tau)}{T_{df}} d\tau + \int_{t_i}^{t_{i+1}} \frac{2(t_{i+1} - \tau)}{T_{df}} e^{-\frac{(t_{i+1} - \tau)}{T_{df}}} d\tau + \int_{t_i}^{t_{i+1}} d\tau$$

$$\left. - \int_{t_i}^{t_{i+1}} 2e^{-\frac{(t_{i+1} - \tau)}{T_{df}}} d\tau + \int_{t_i}^{t_{i+1}} e^{-2\frac{(t_{i+1} - \tau)}{T_{df}}} d\tau \right] \quad (B-6)$$

After performing the indicated integrations and combining terms, the result is:

$$\int_{t_i}^{t_{i+1}} A^2(\tau) Q_{11}(\tau) d\tau = \sigma_{df}^2 \left[\frac{(2)(T_{df})(\Delta t)^3}{3} - (2)(T_{df})^2(\Delta t)^2 \right.$$

$$\left. - (4)(T_{df})^3(\Delta t) \left(e^{-\frac{\Delta t}{T_{df}}} \right) + (2)(T_{df})^3(\Delta t) + (T_{df})^4 \left(e^{-\frac{2\Delta t}{T_{df}}} \right) + T_{df}^4 \right] \quad (B-7)$$

The remainder of the terms of Equation (B-5) are evaluated in a similiar manner with the results being as expressed in Equation (3-12).

Appendix C

Plots of Tracking Errors

This appendix gives the sequence of plots described in Chapter 5. The figures are numbered so as to correspond directly to the case numbers used in Chapter 5, i.e. Figures C-2a thru C-2j are the plots which correspond to case 2 in Chapter 5. For a description of the truth model and filter parameters refer to the appropriate table in Chapter 5 or the summary table which is given in section 5.16. (Note: In order for the case and figure numbers to correspond, Figure C-1 is not used). For the cases where the filter tuning plots are not shown, the tuning employed was similar to case 1 for single and case 3 for multiple hot spot targets.

FILTER VS ACTUAL ERROR (X-POSITION)

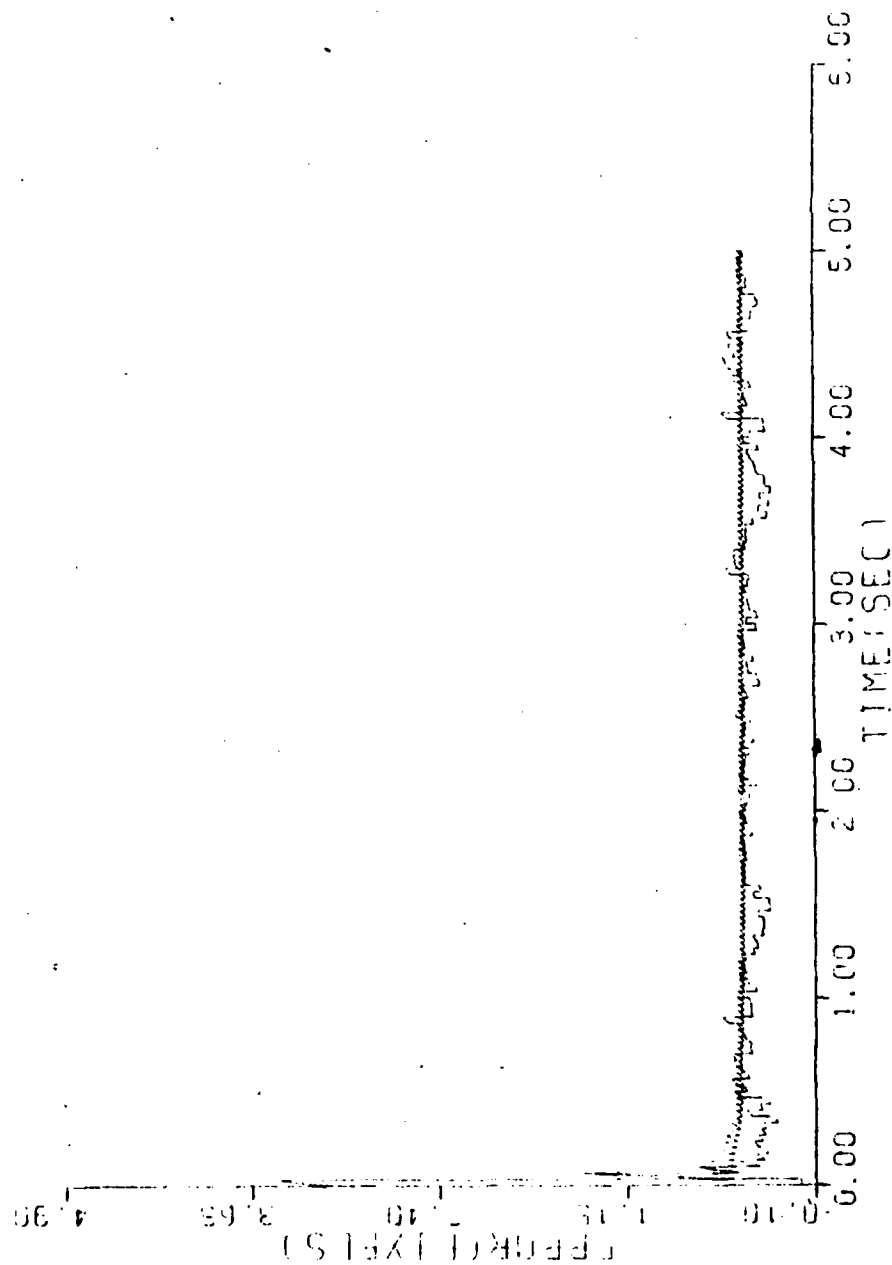


Figure C-2a. Case 2

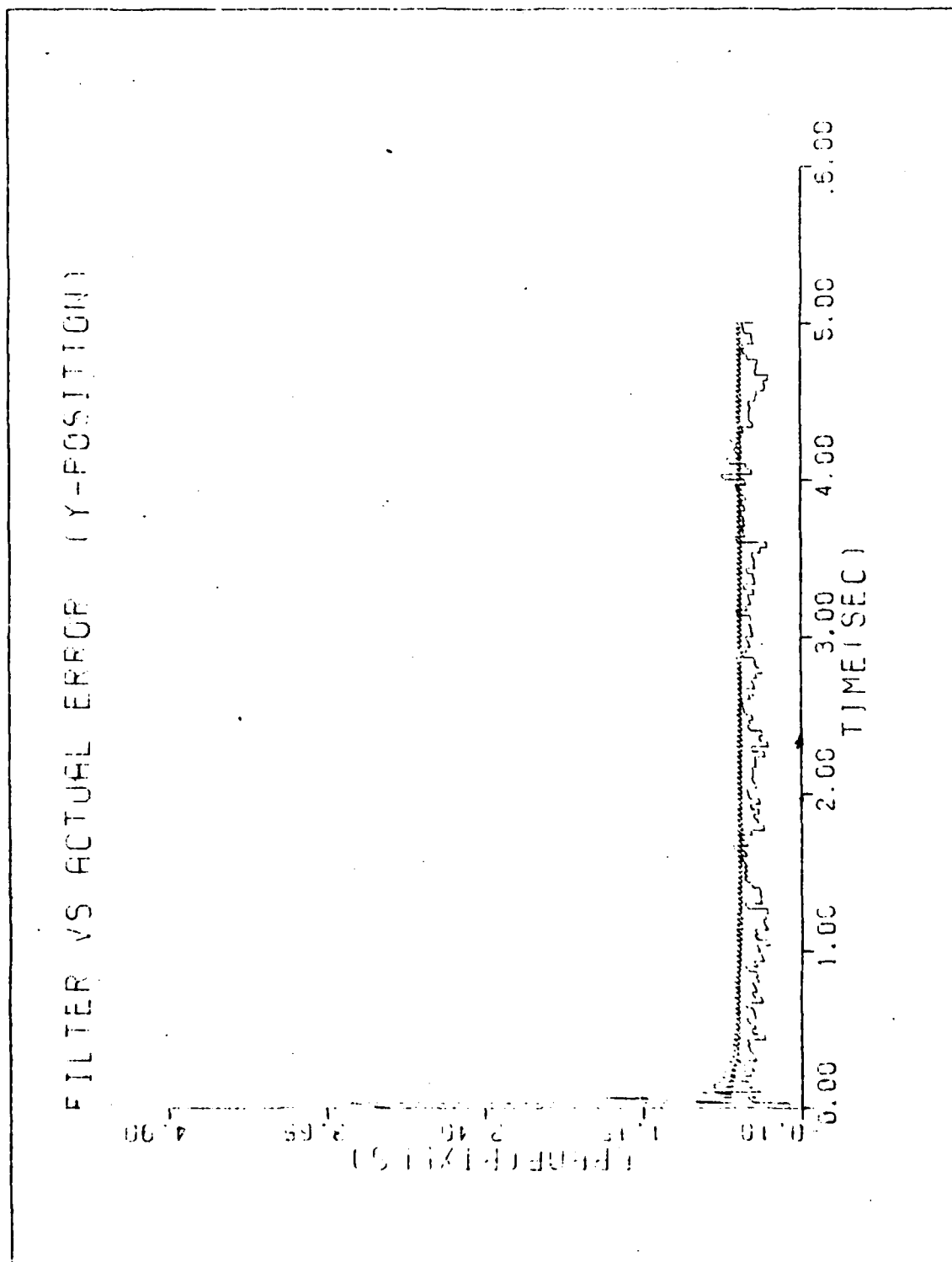


Figure C-2b. Case 2

ESTIMATED X-MINUS POSITION (+/-) SIGMA

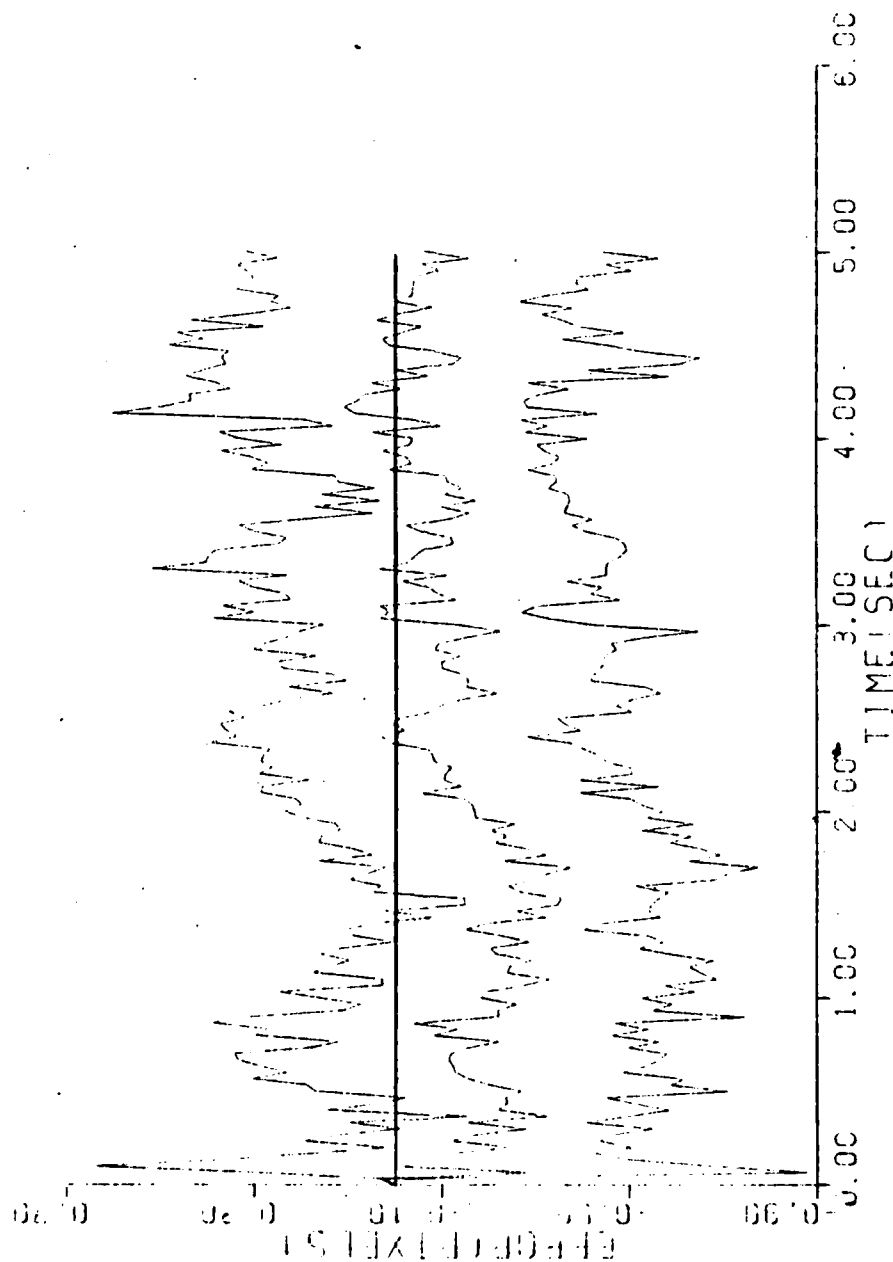


Figure C-2c. Case 2

ESTIMATED I-MINUS POSITION (+/-) SIGMA

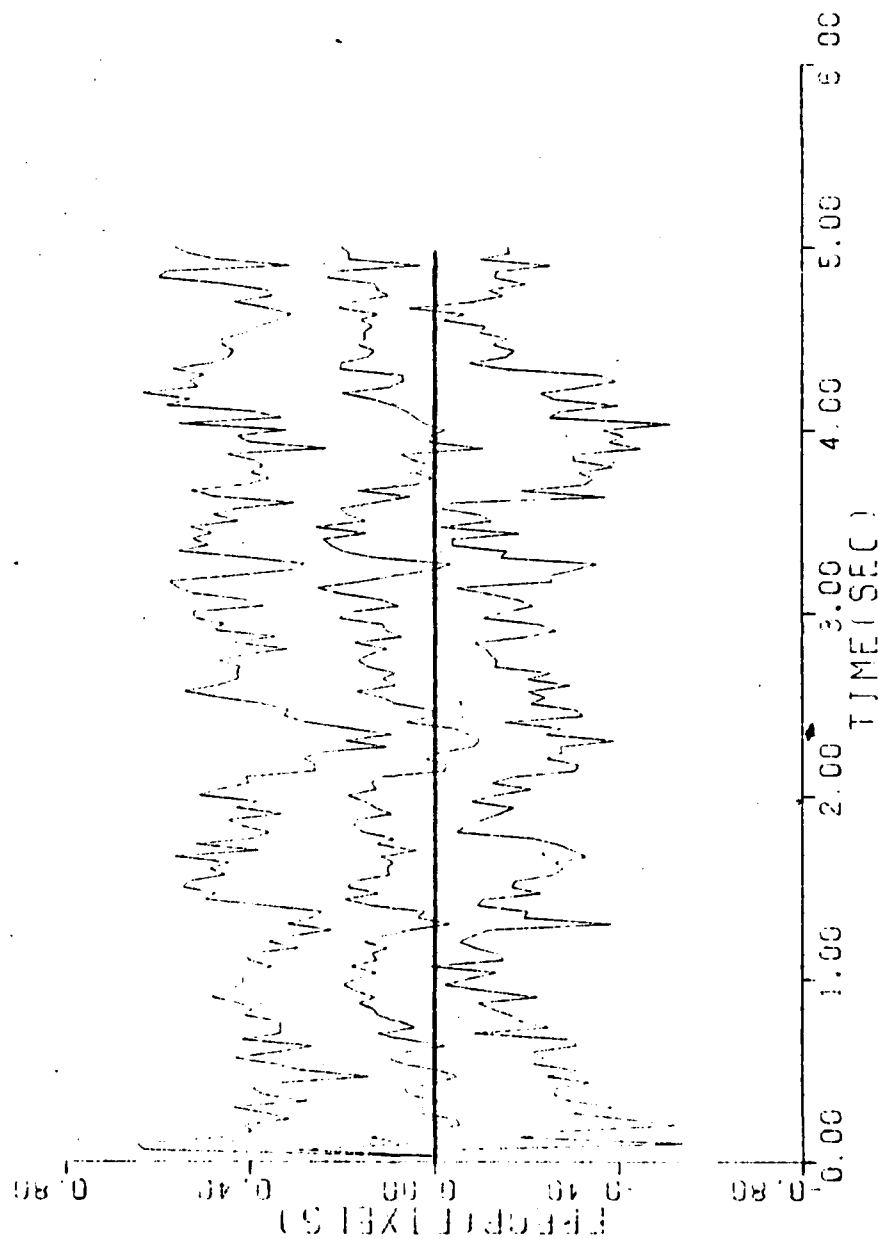


Figure C-2d. Case 2

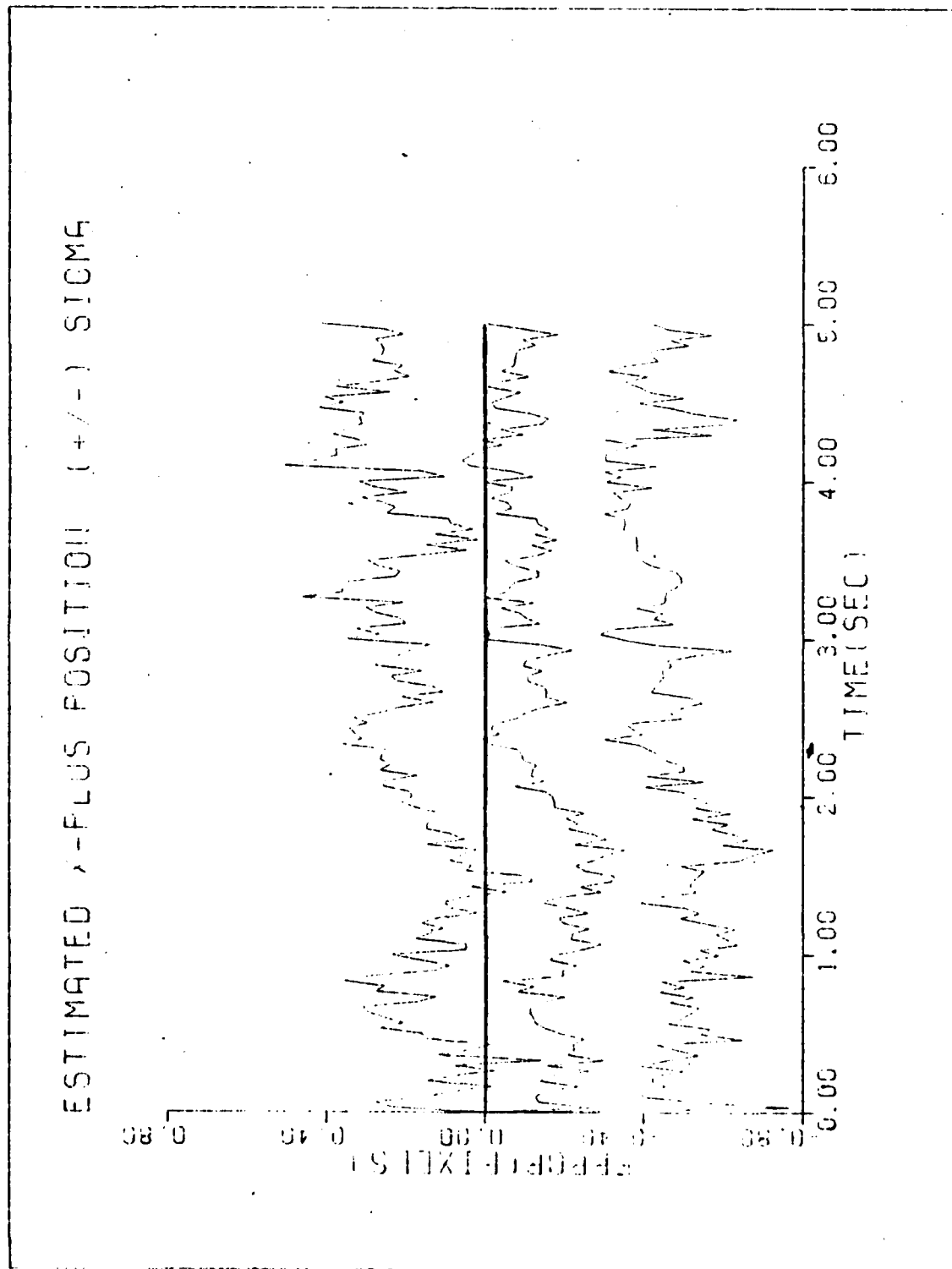


Figure C-2e. Case 2

ESTIMATED 1-PLUS POSITION (+/-) SIGMA

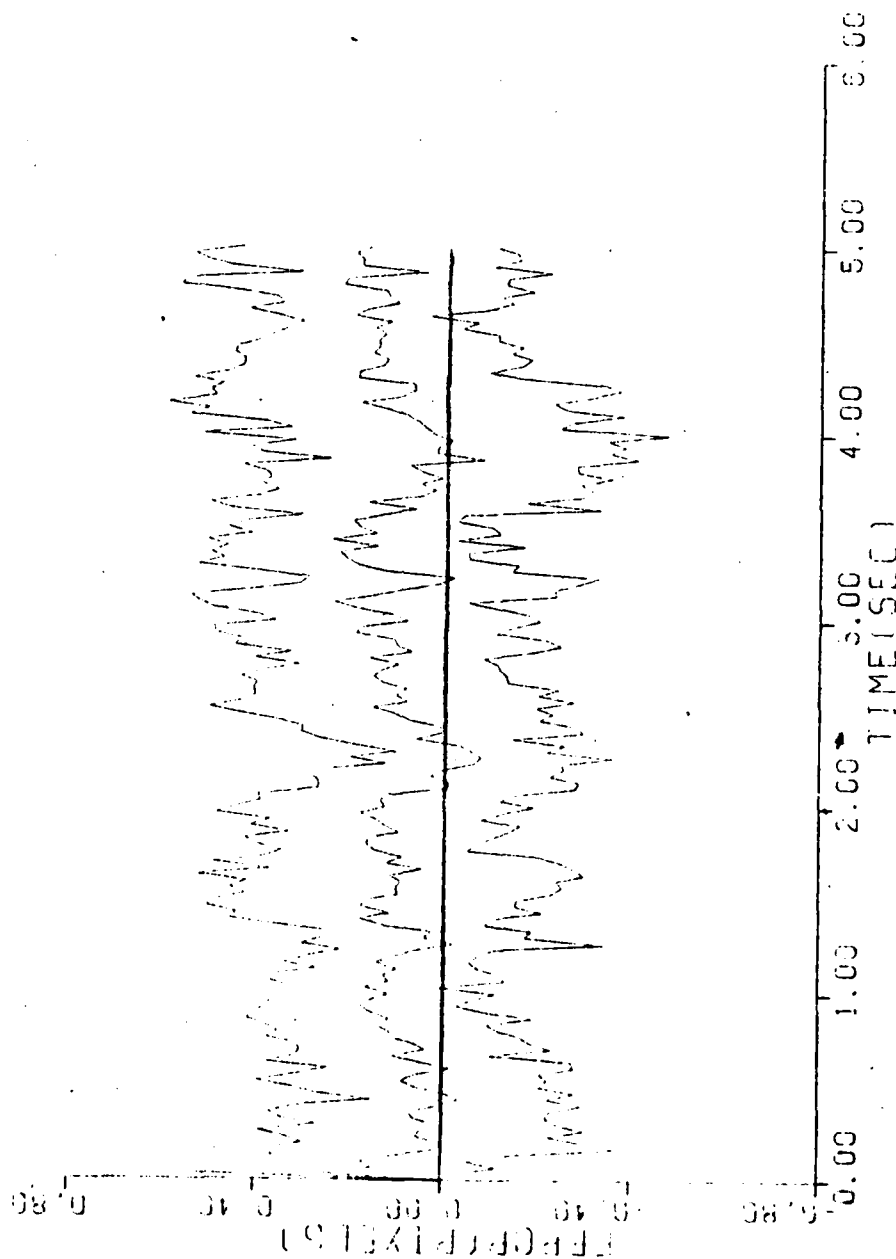


Figure C-2f. Case 2

ESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA

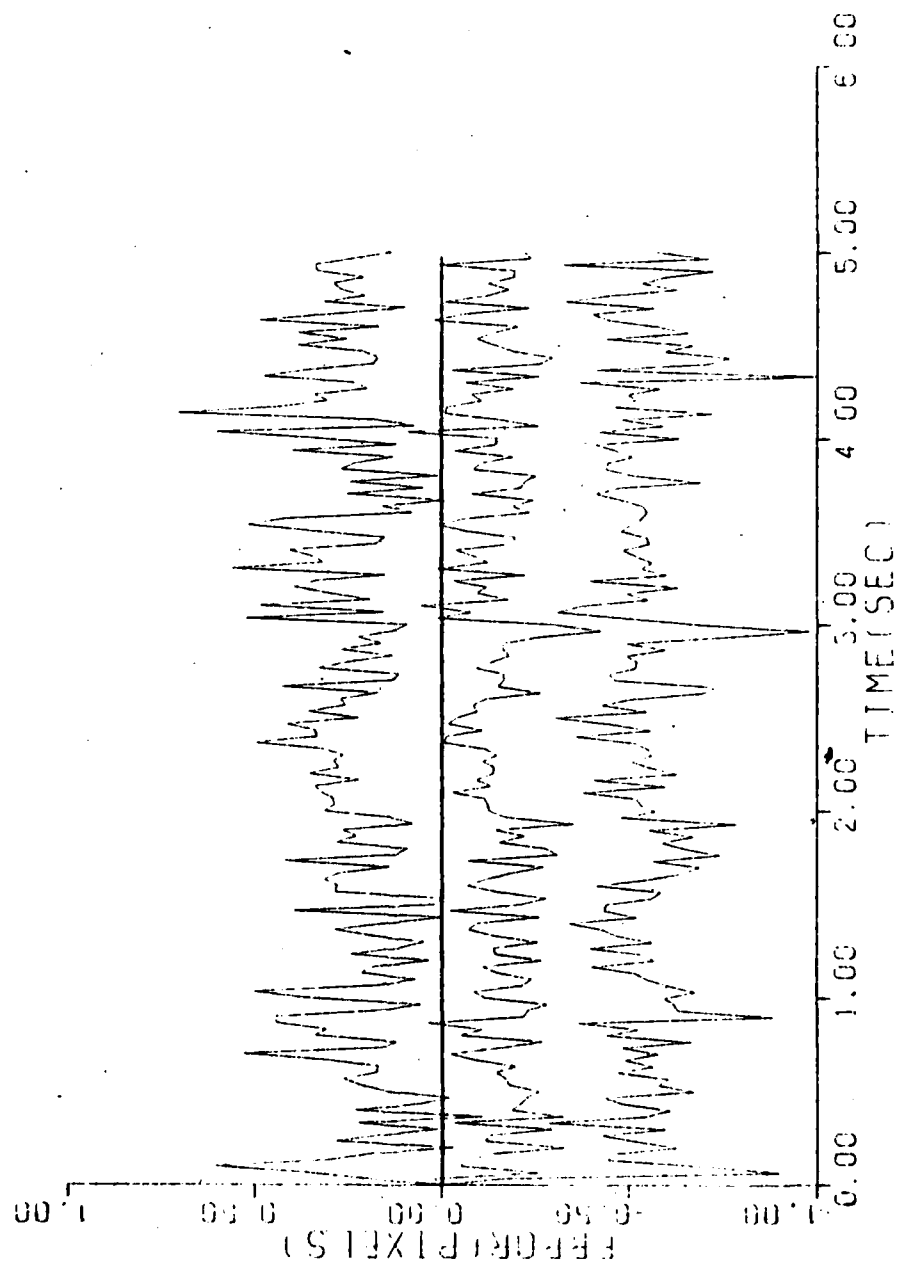


Figure C-2g. Case 2

ESTIMATED Y-MINUS CENTROID POSITION (---) SIGMA

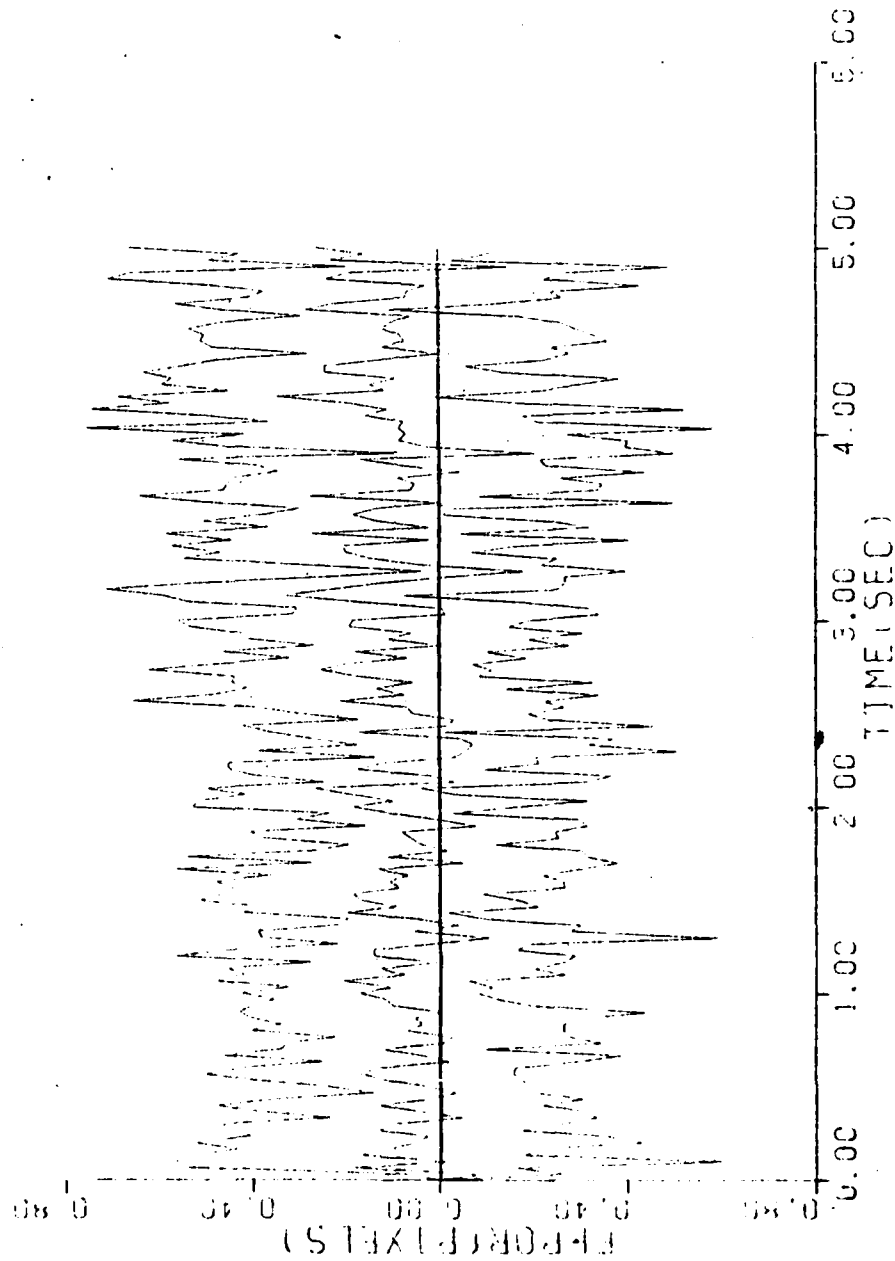


Figure C-2h. Case 2

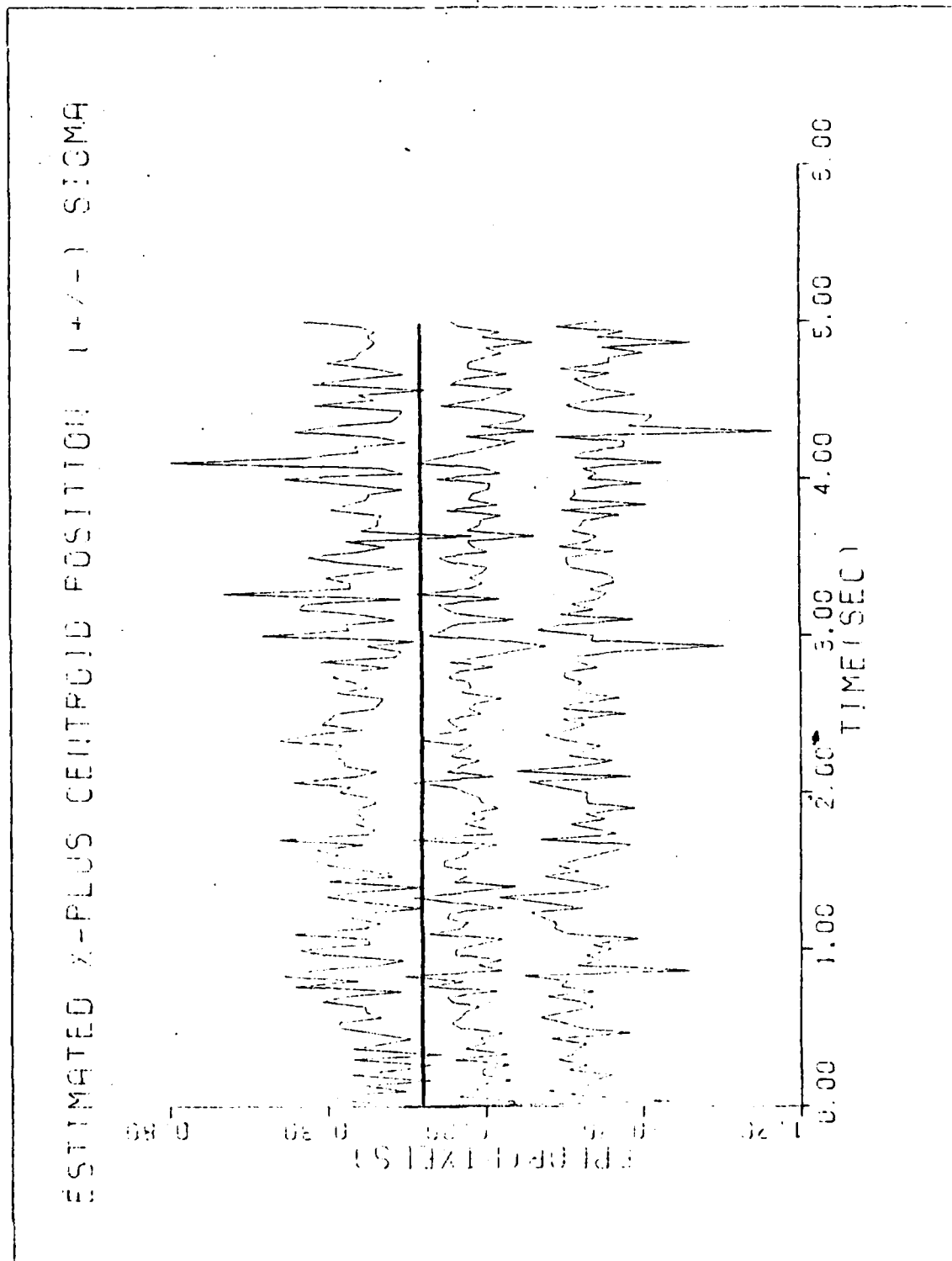


Figure C-2i. Case 2

AD-A124 884

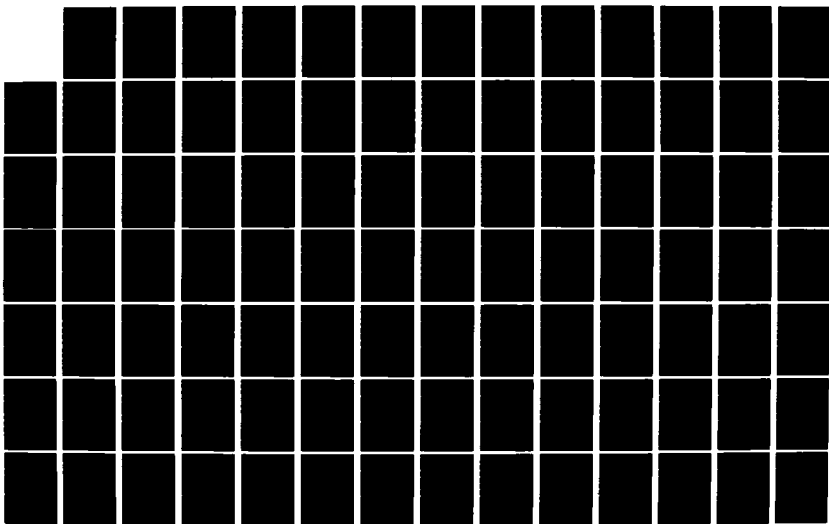
ENHANCED TRACKING OF AIRBORNE TARGETS USING A
CORRELATOR/KALMAN FILTER(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING
P P MILLNER DEC 82 AFIT/GE/EE/82D-58

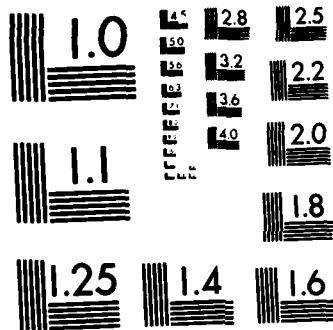
3/4

UNCLASSIFIED

F/G 12/1

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

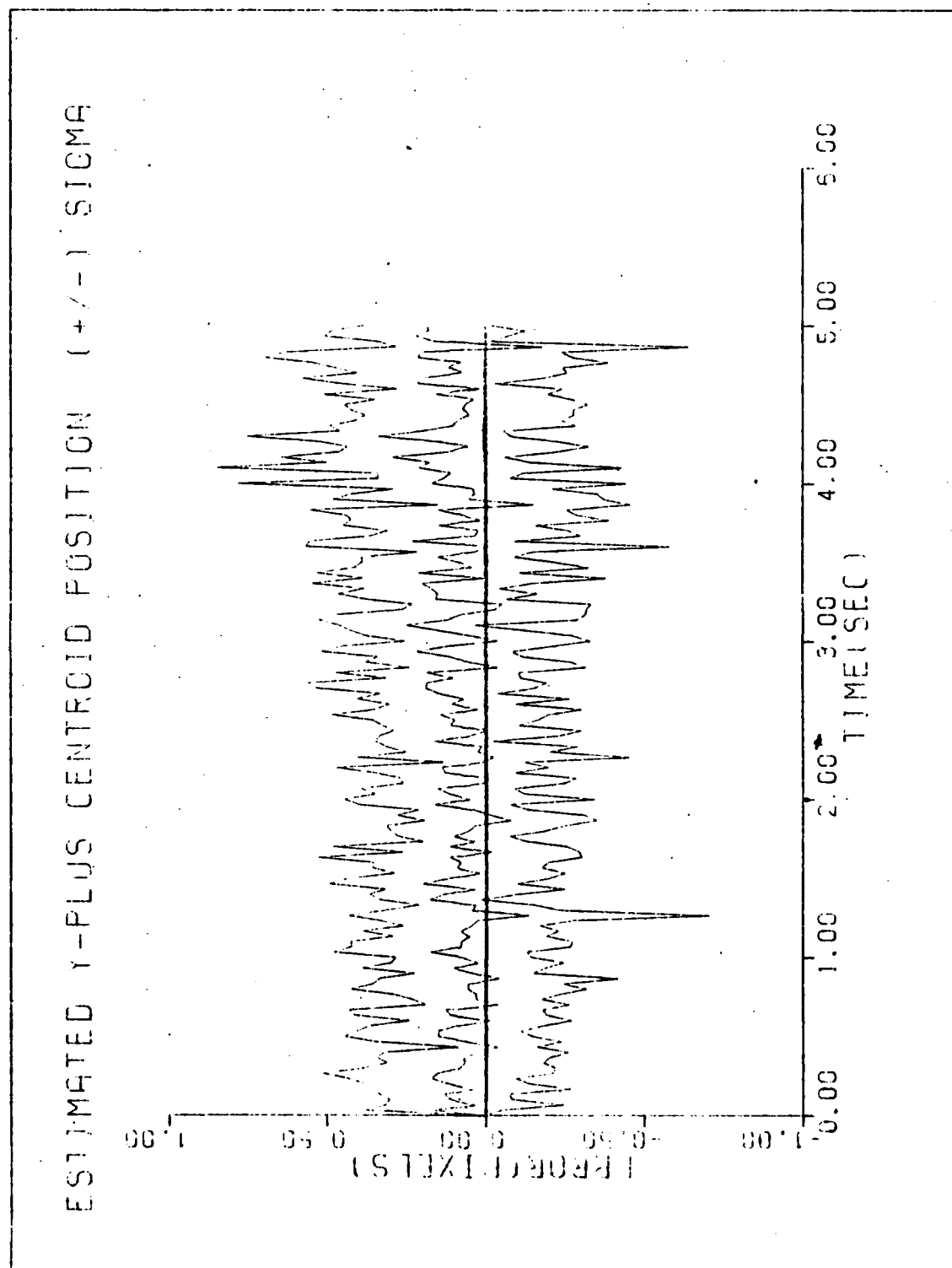


Figure C-2j. Case 2

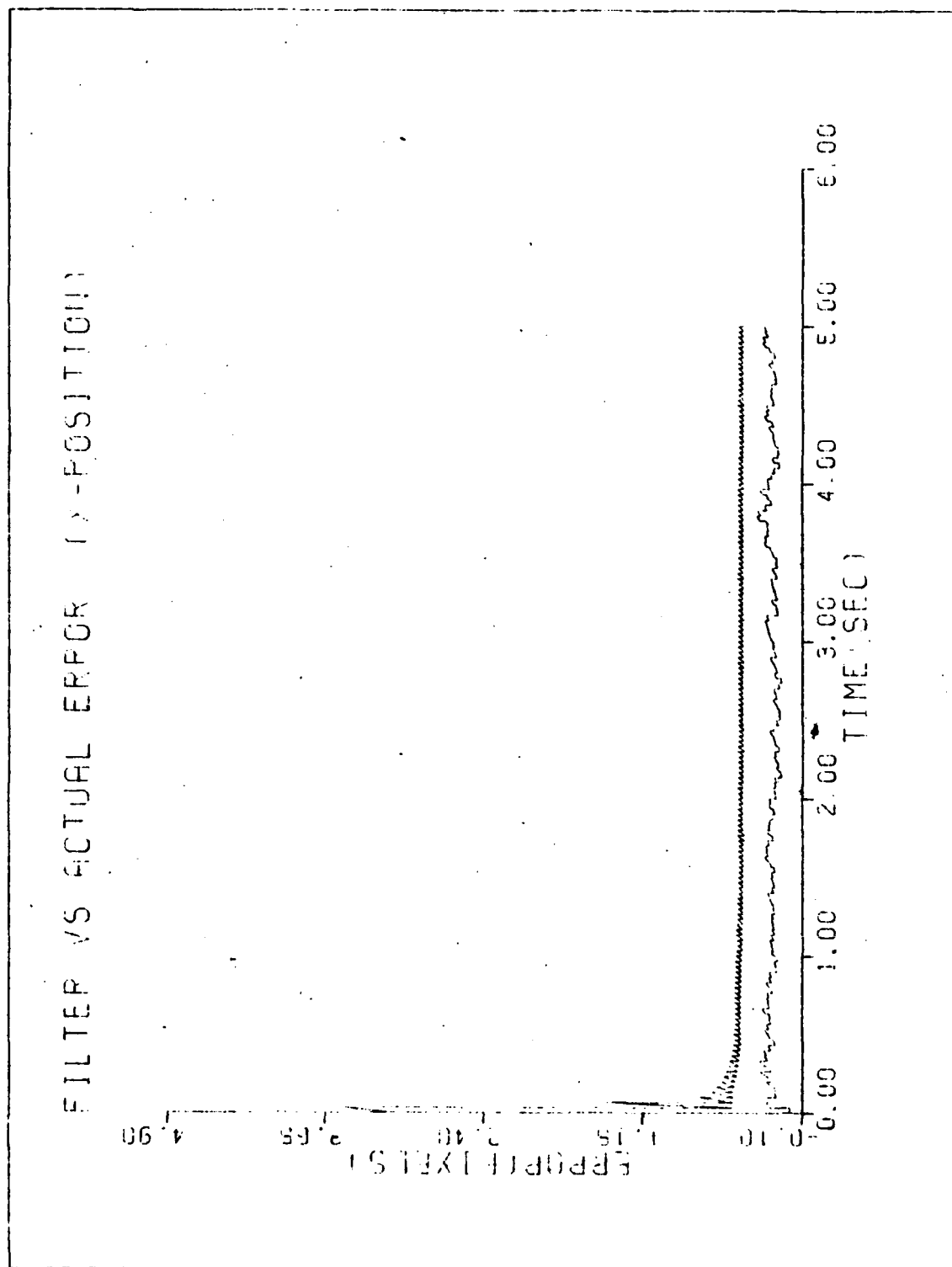


Figure C-3a. Case 3

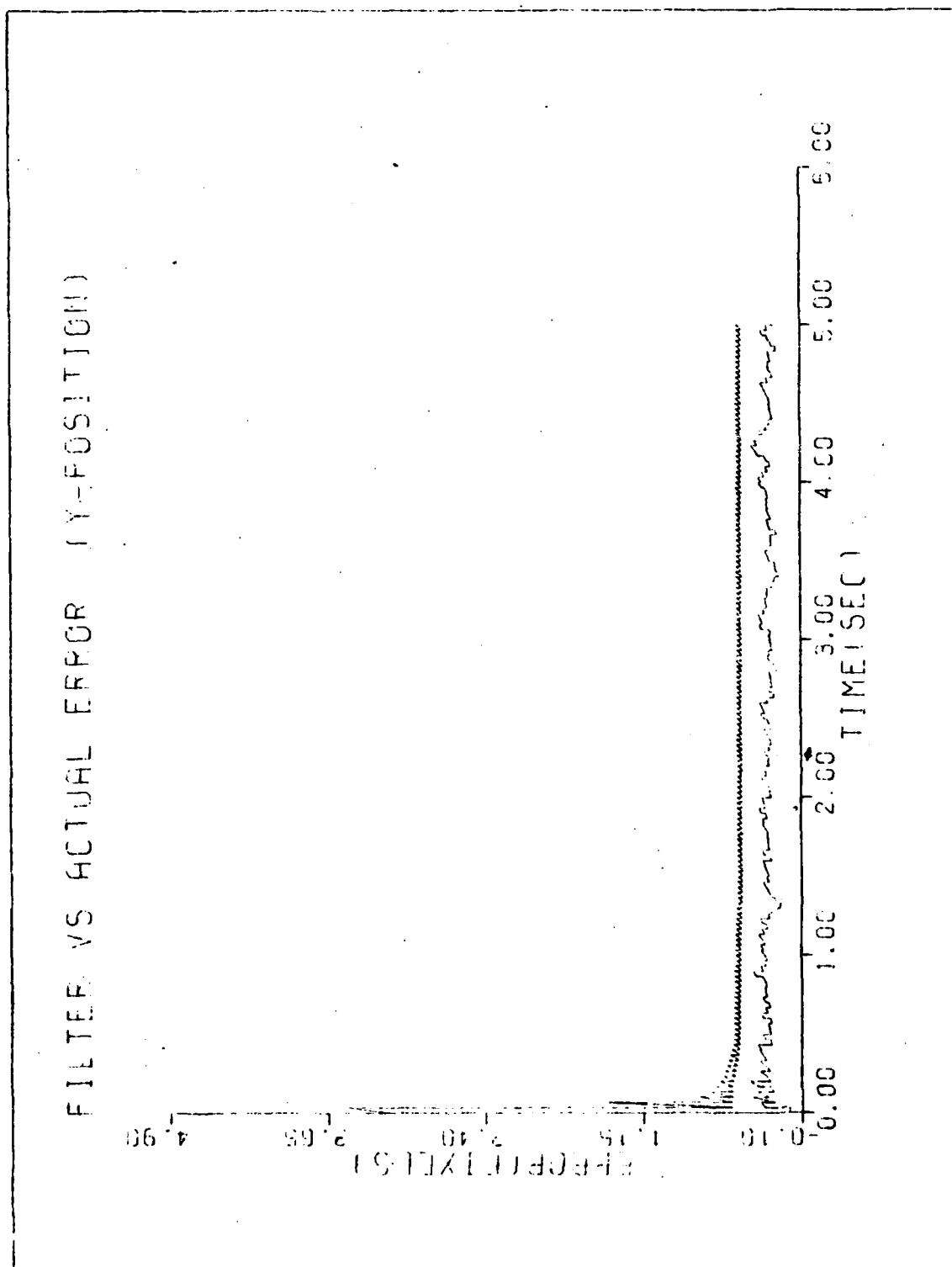


Figure C-3b. Case 3

ESTIMATED X-MINUS POSITION (+/-) SIGMA

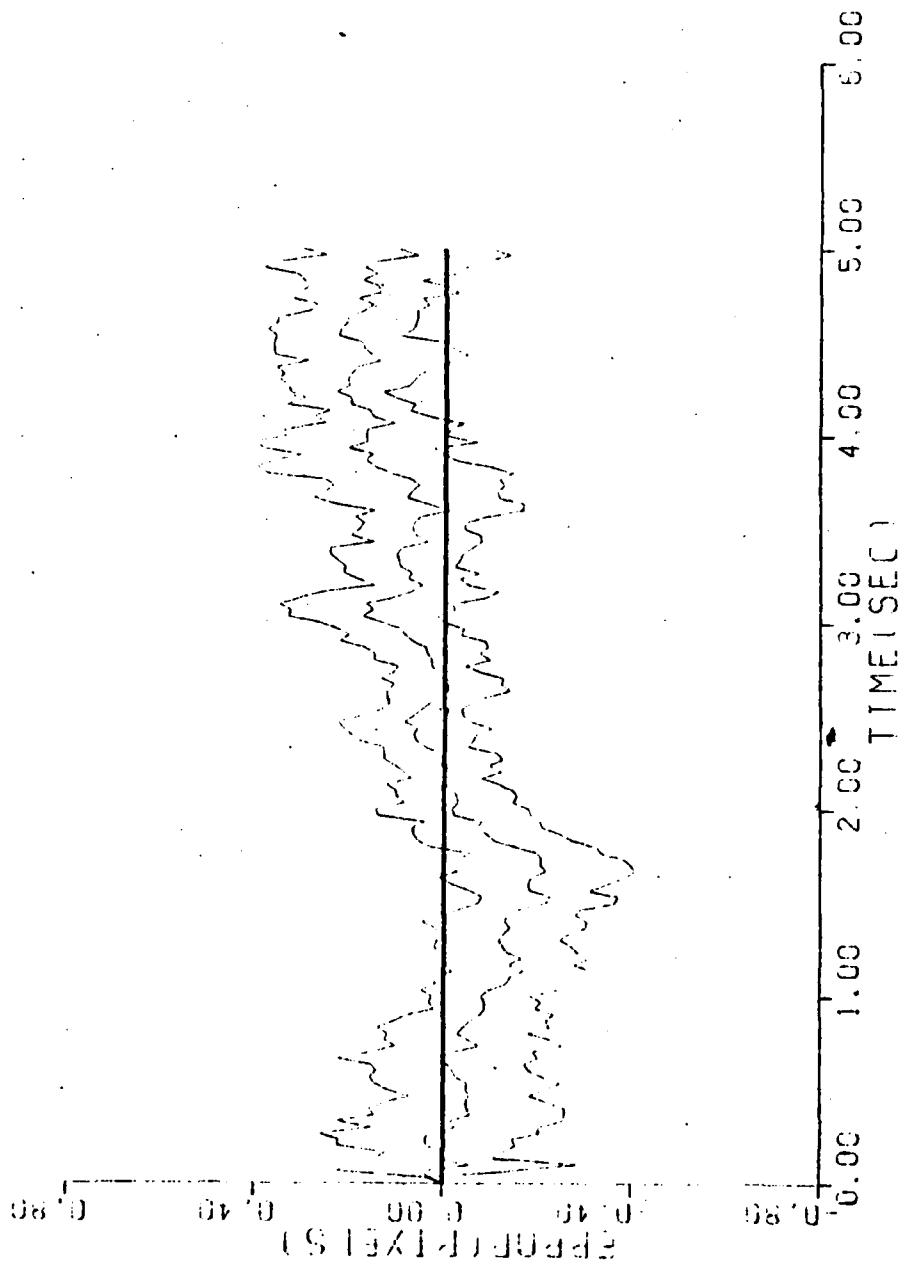


Figure C-3c. Case 3

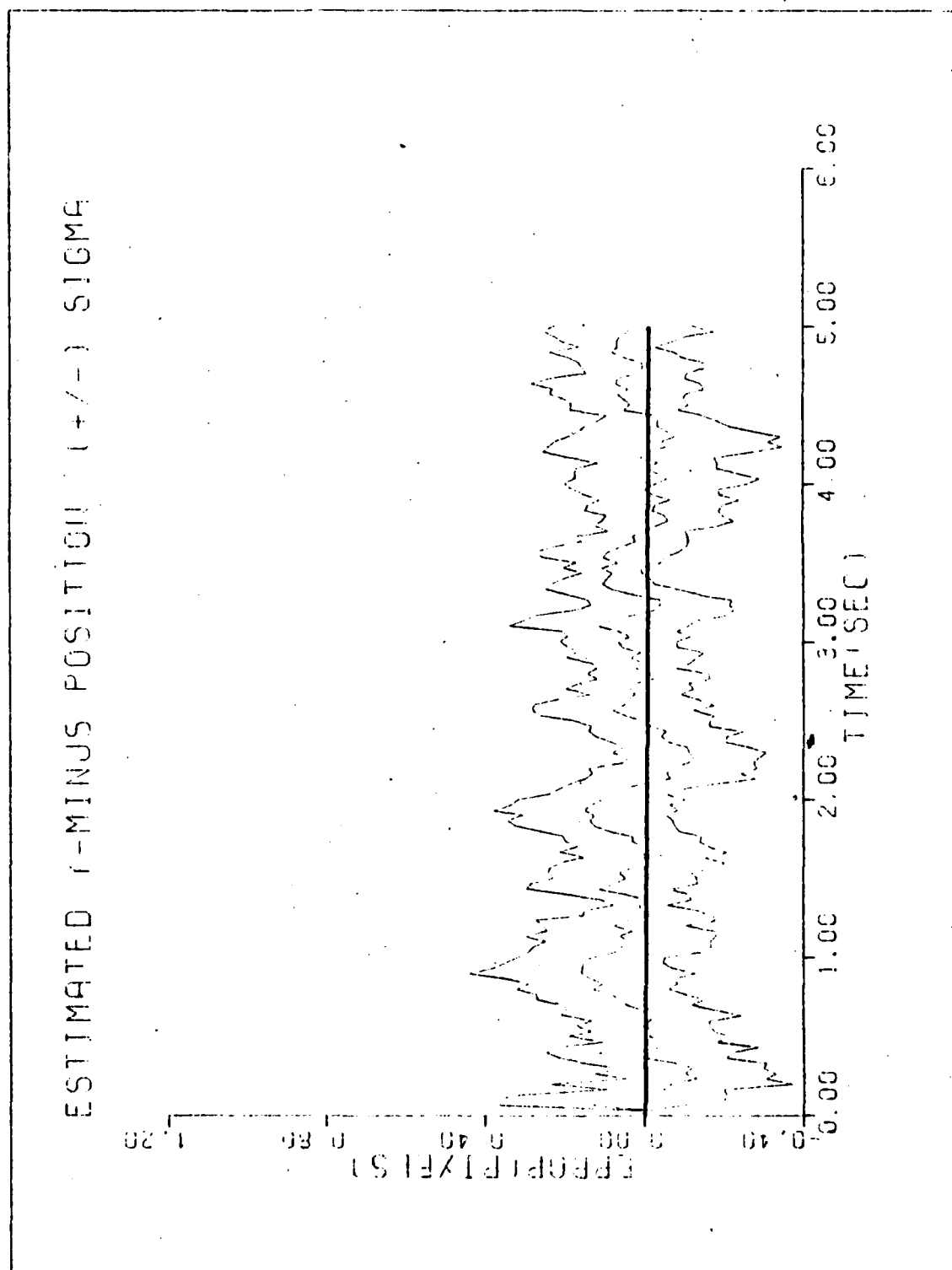


Figure C-3d. Case 3

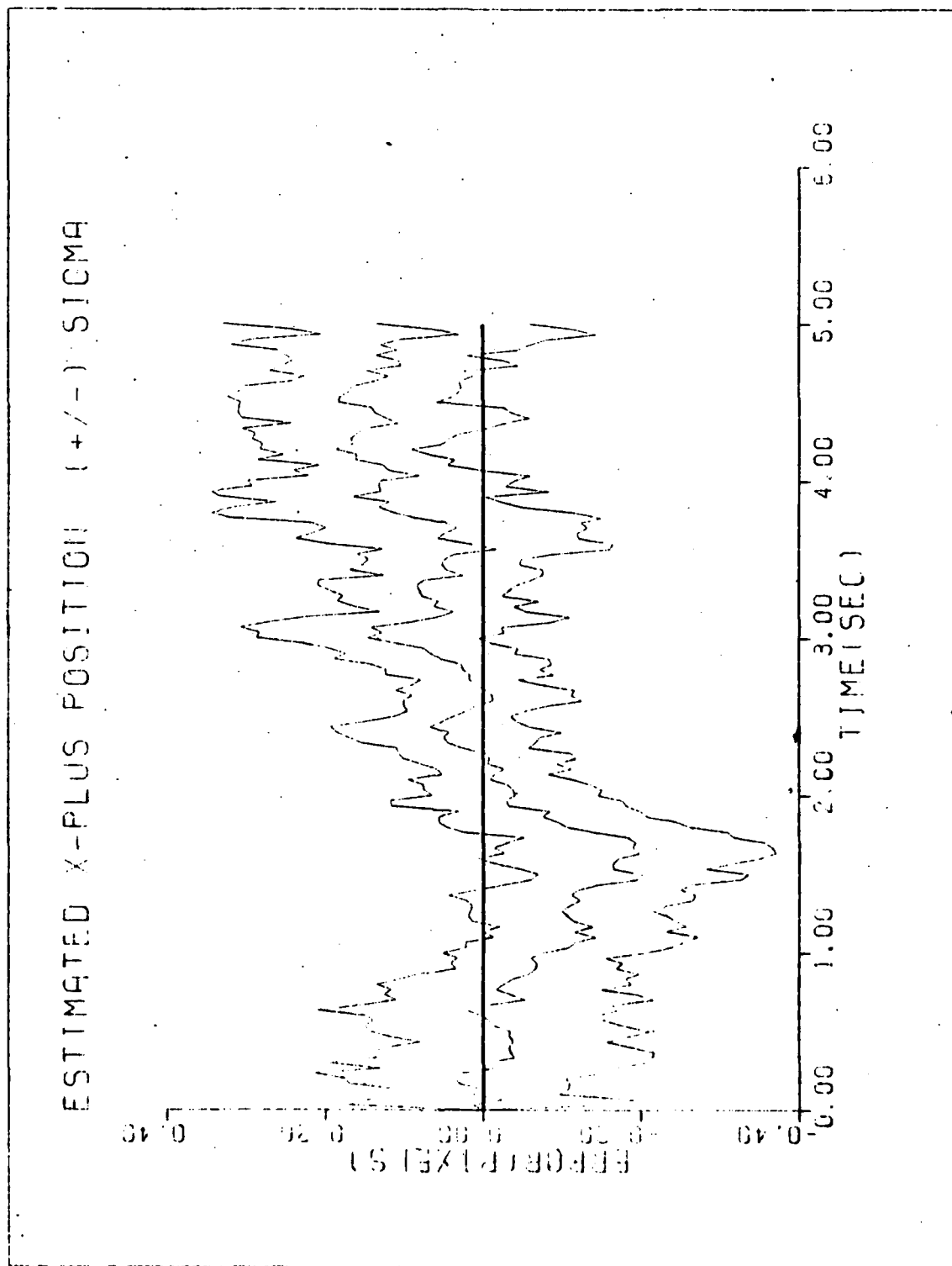


Figure C-3e. Case 3

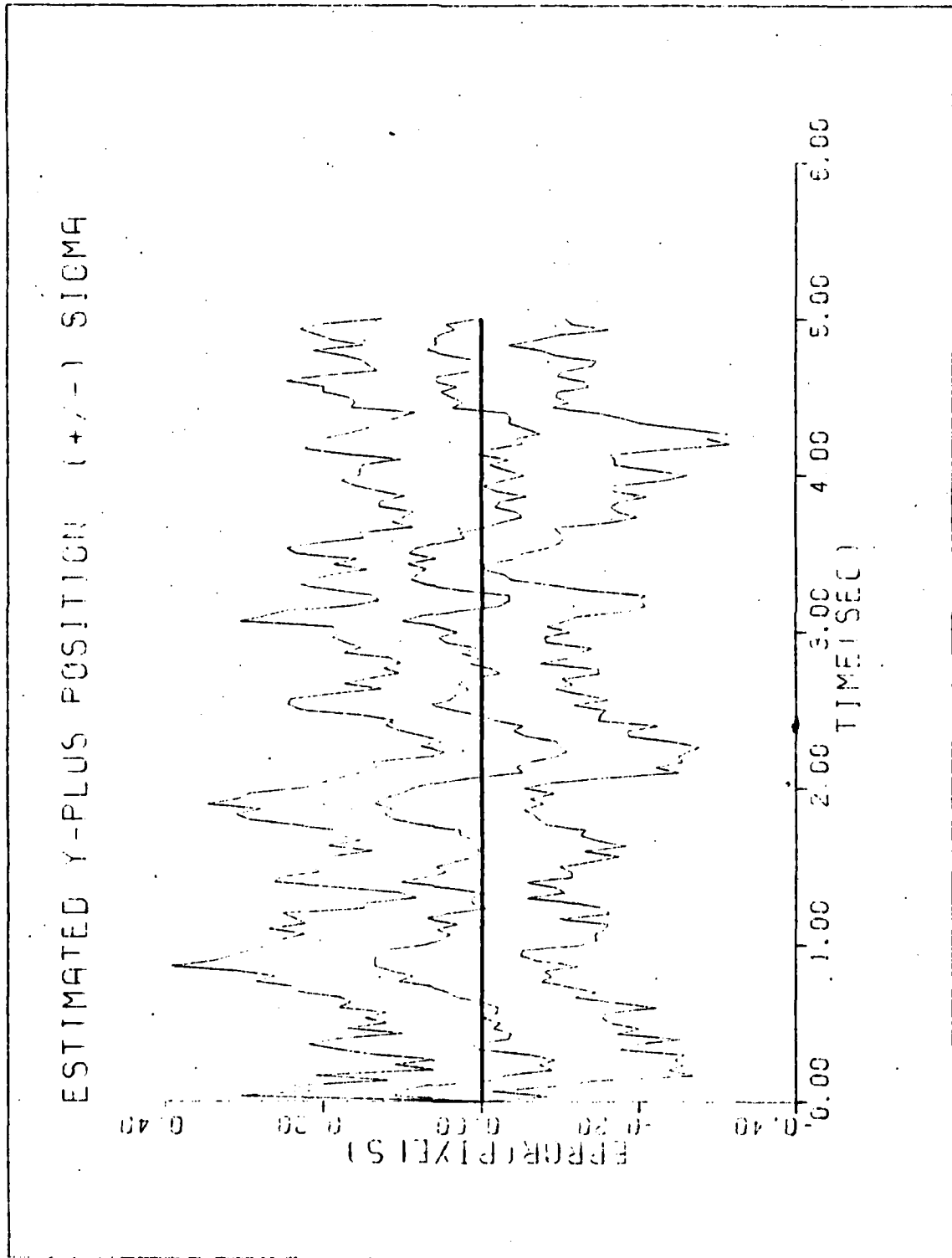


Figure C-3f. Case 3

ESTIMATED X-MINUS CENTROID POSITION. (+/-) SIGMA

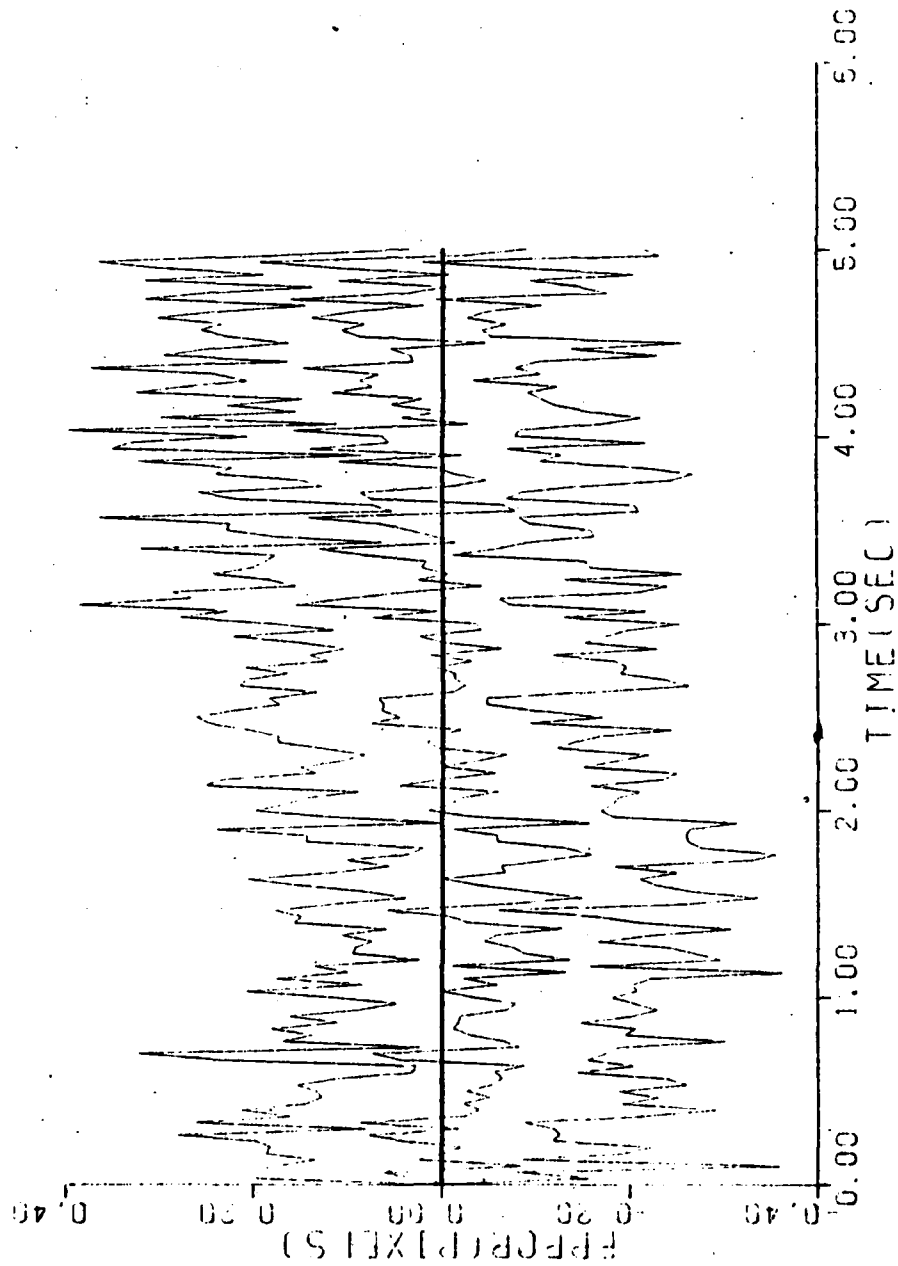


Figure C-3g. Case 3

B

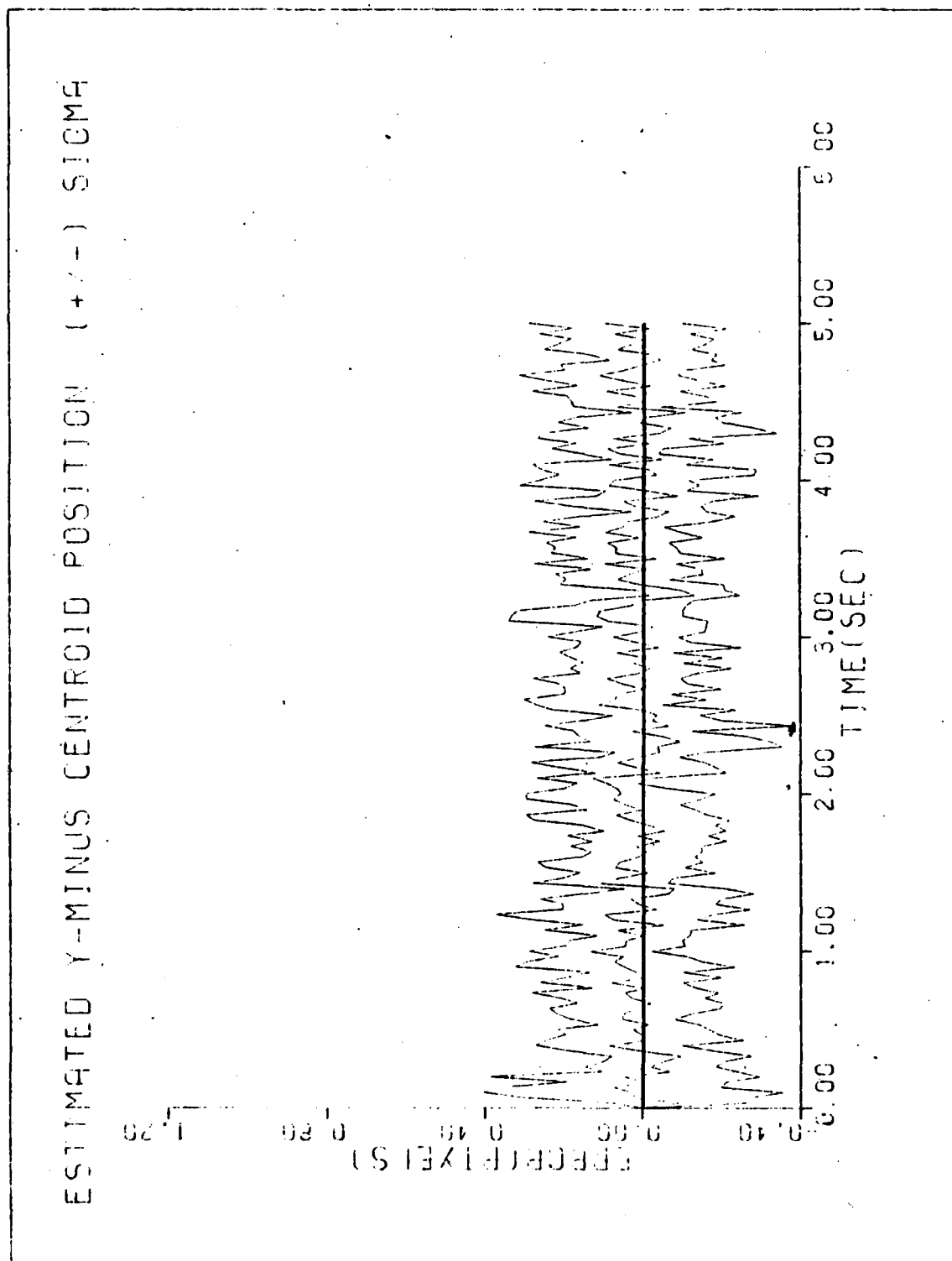


Figure C-3h. Case 3

ESTIMATED X-PLUS CENTROID POSITION (+/-) SIGMA

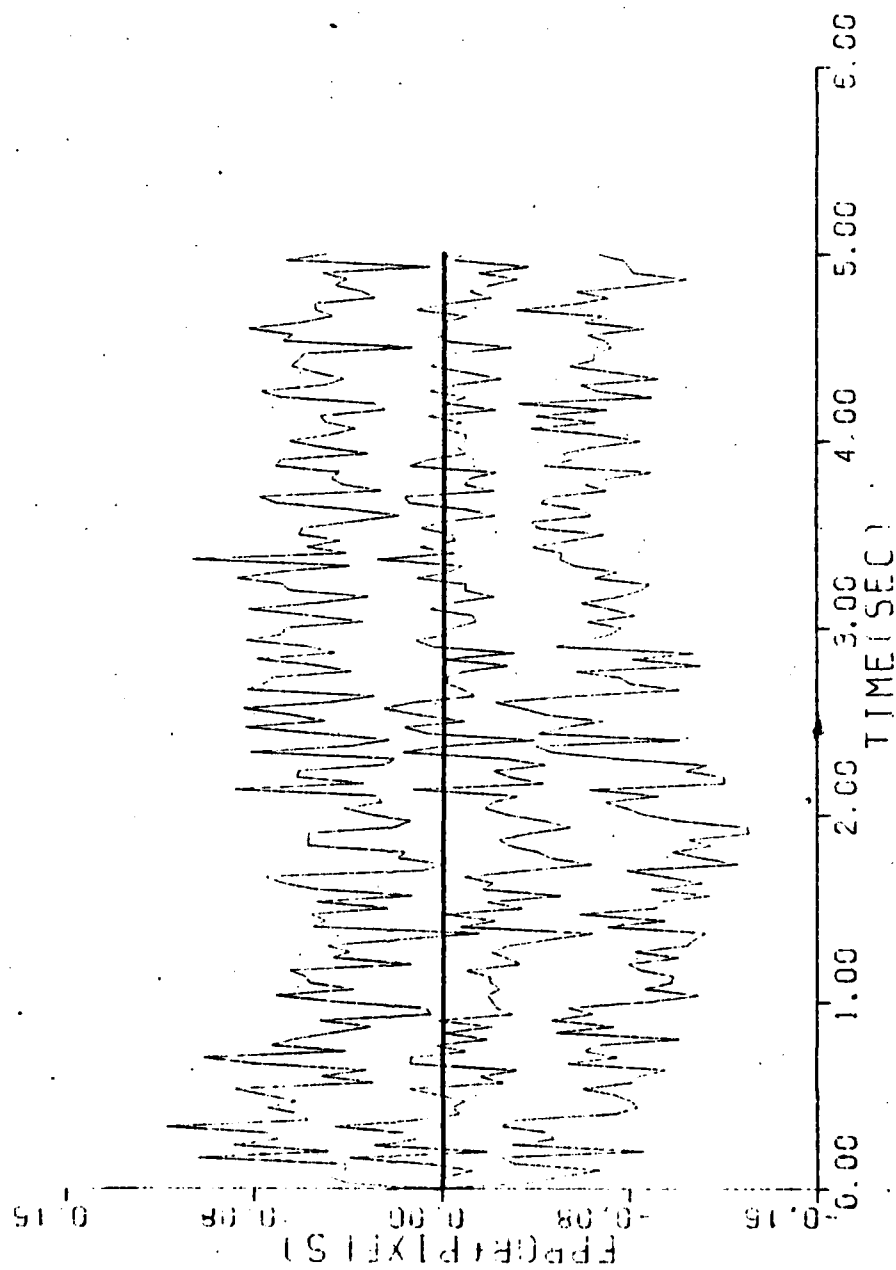


Figure C-31. Case 3

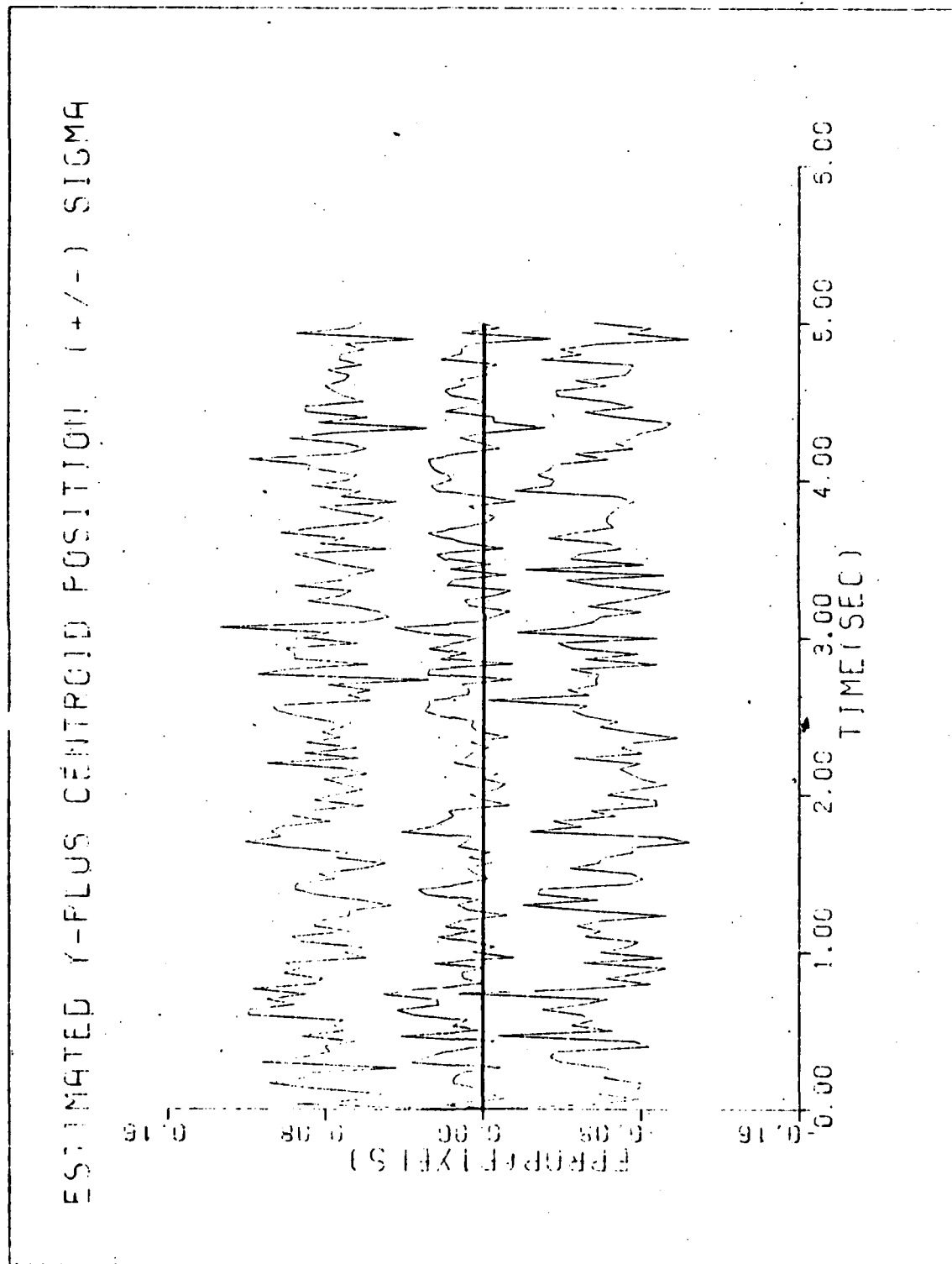


Figure C-3j. Case 3

FILTER VS ACTUAL ERROR (X-POSITION)

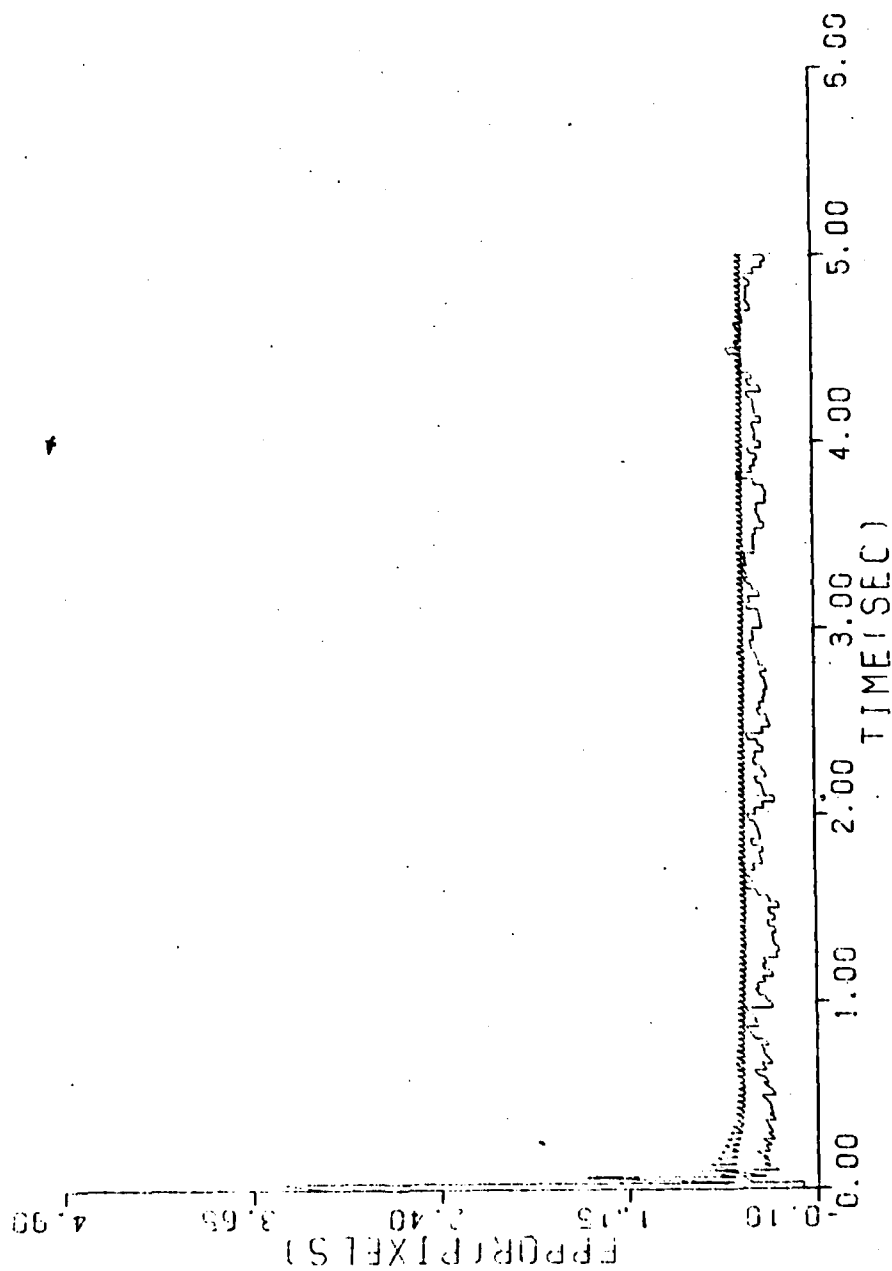


Figure C-4a. Case 4

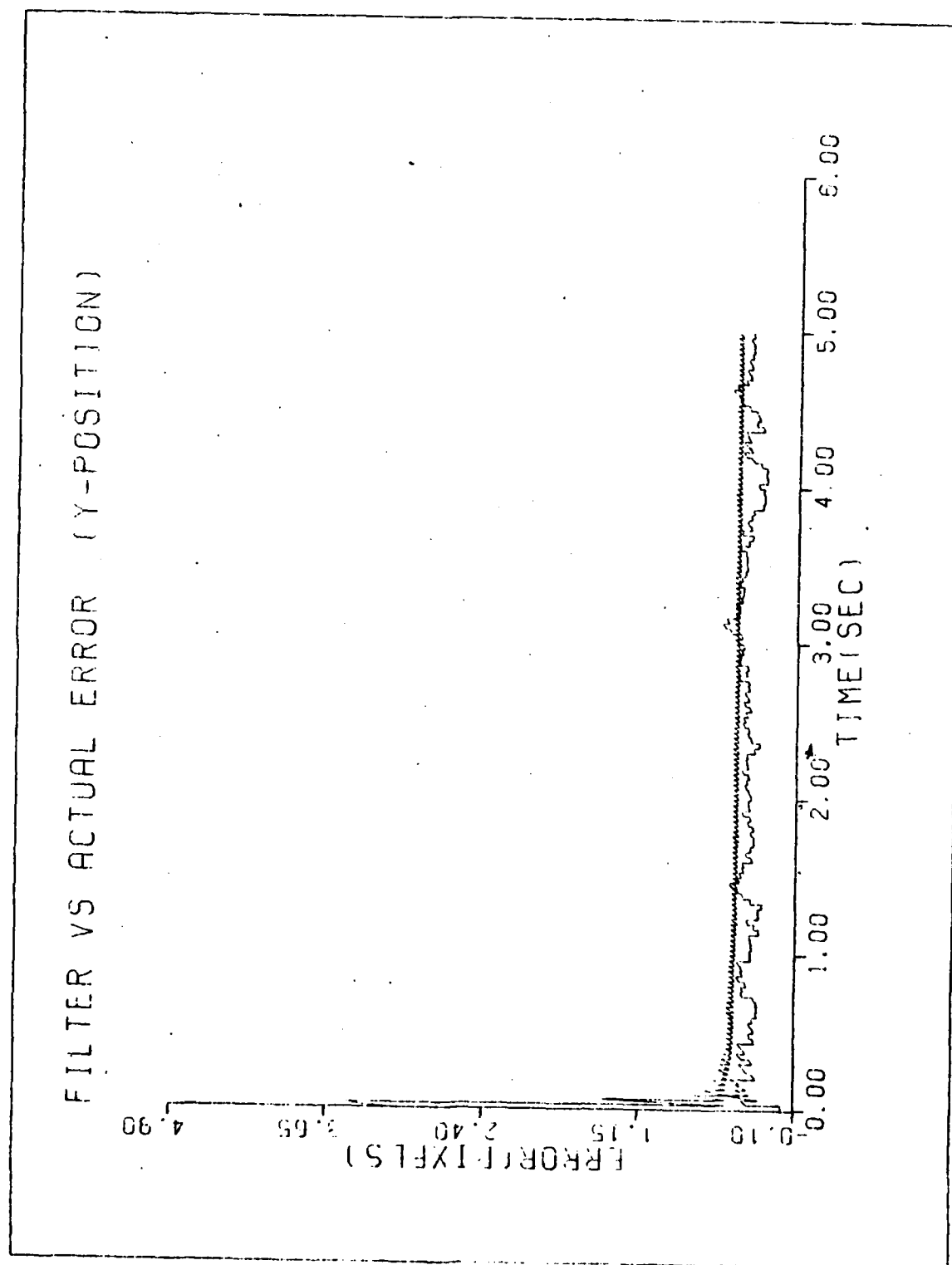


Figure C-4b. Case 4

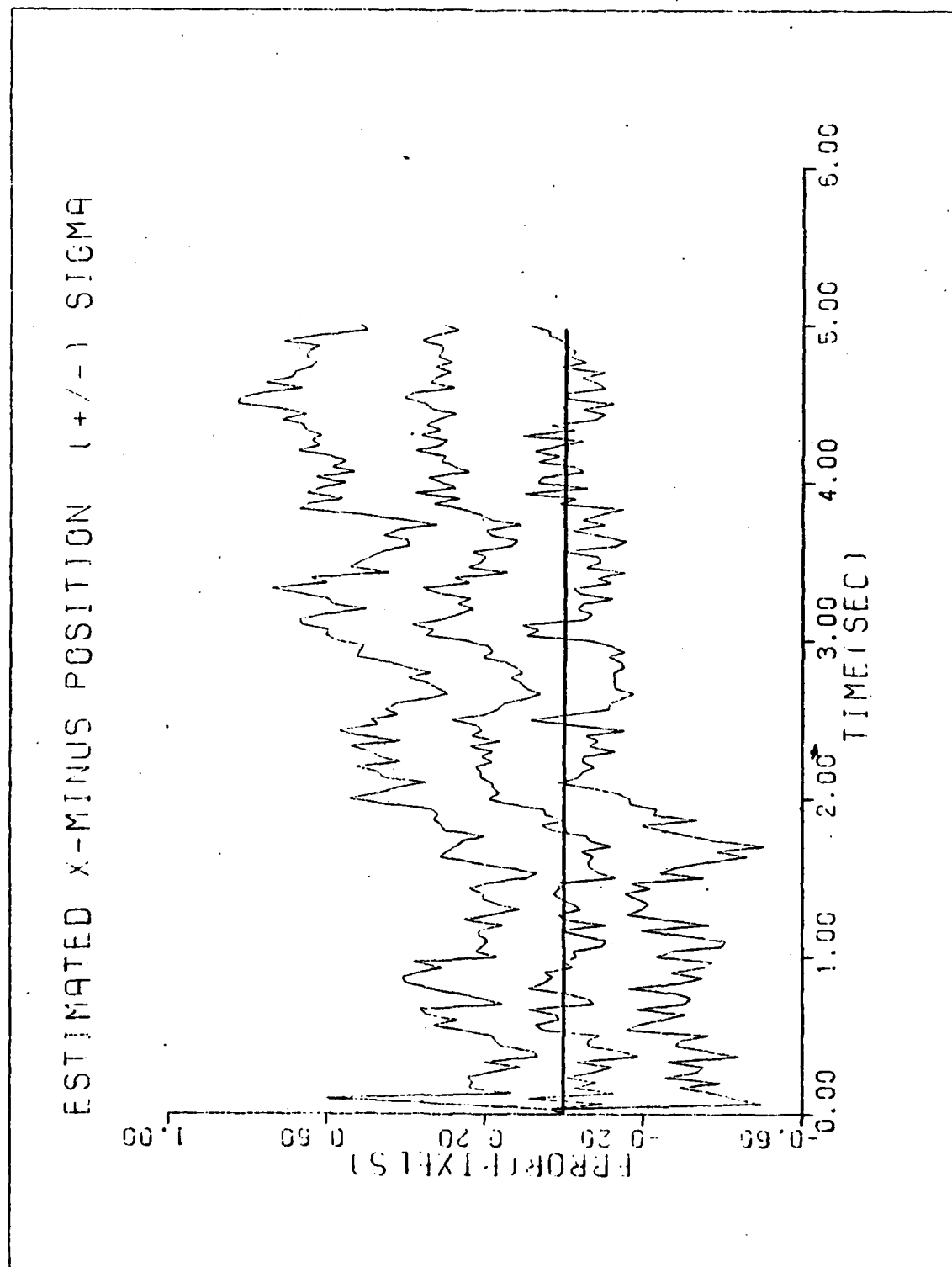


Figure C-4c. Case 4

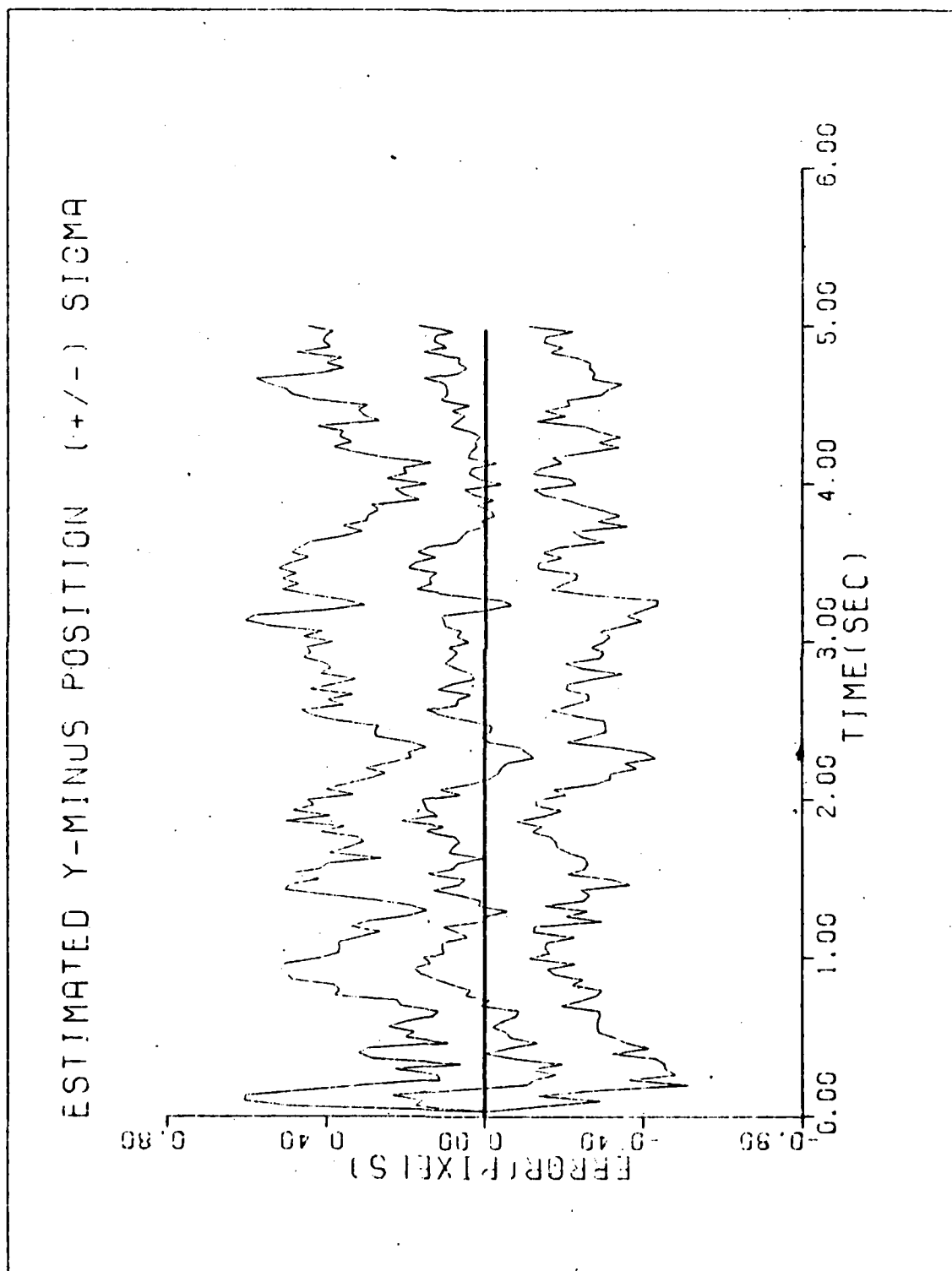


Figure C-4d. Case 4

ESTIMATED X-PLUS POSITION (+/-) SIGMA

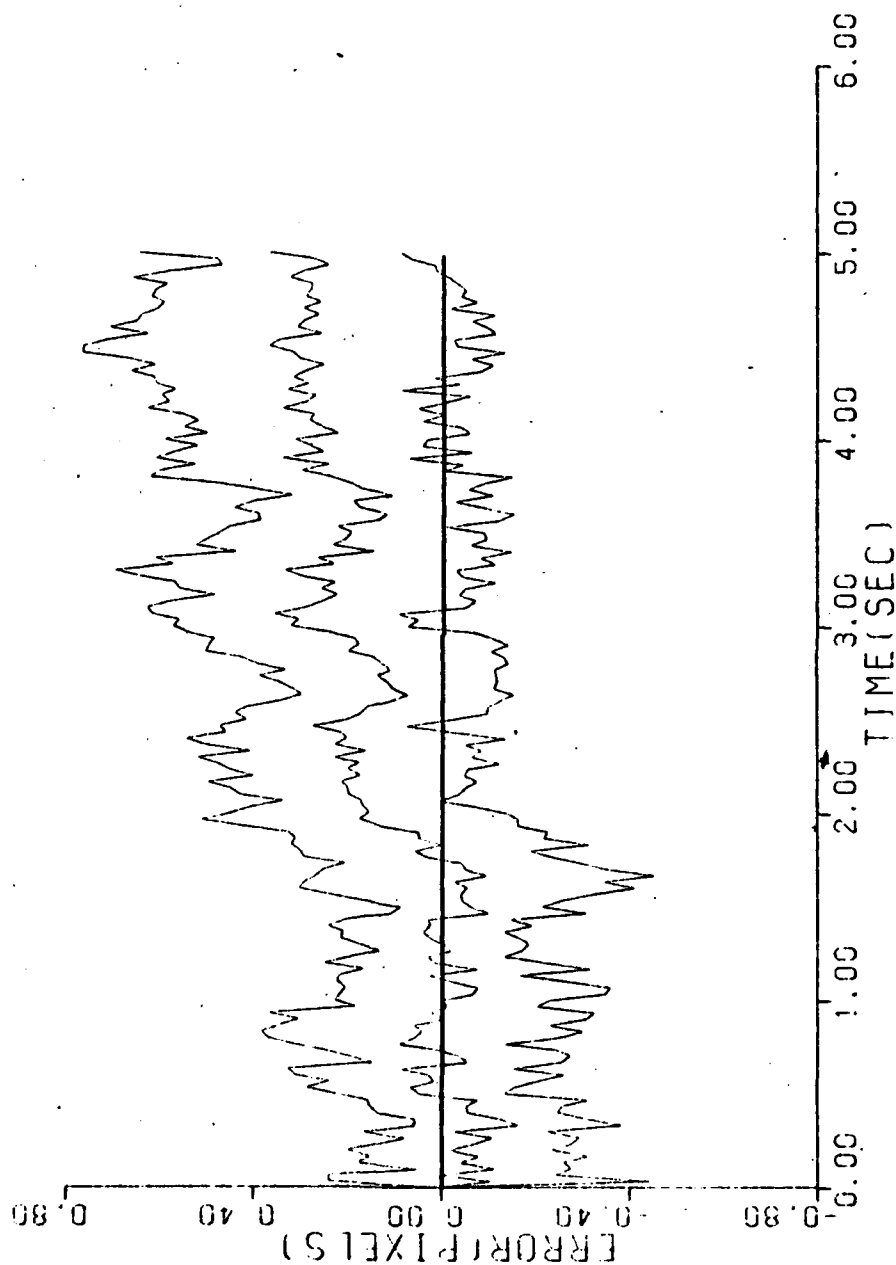


Figure C-4e. Case 4

ESTIMATED Y-PLUS POSITION (+/-) SIGMA

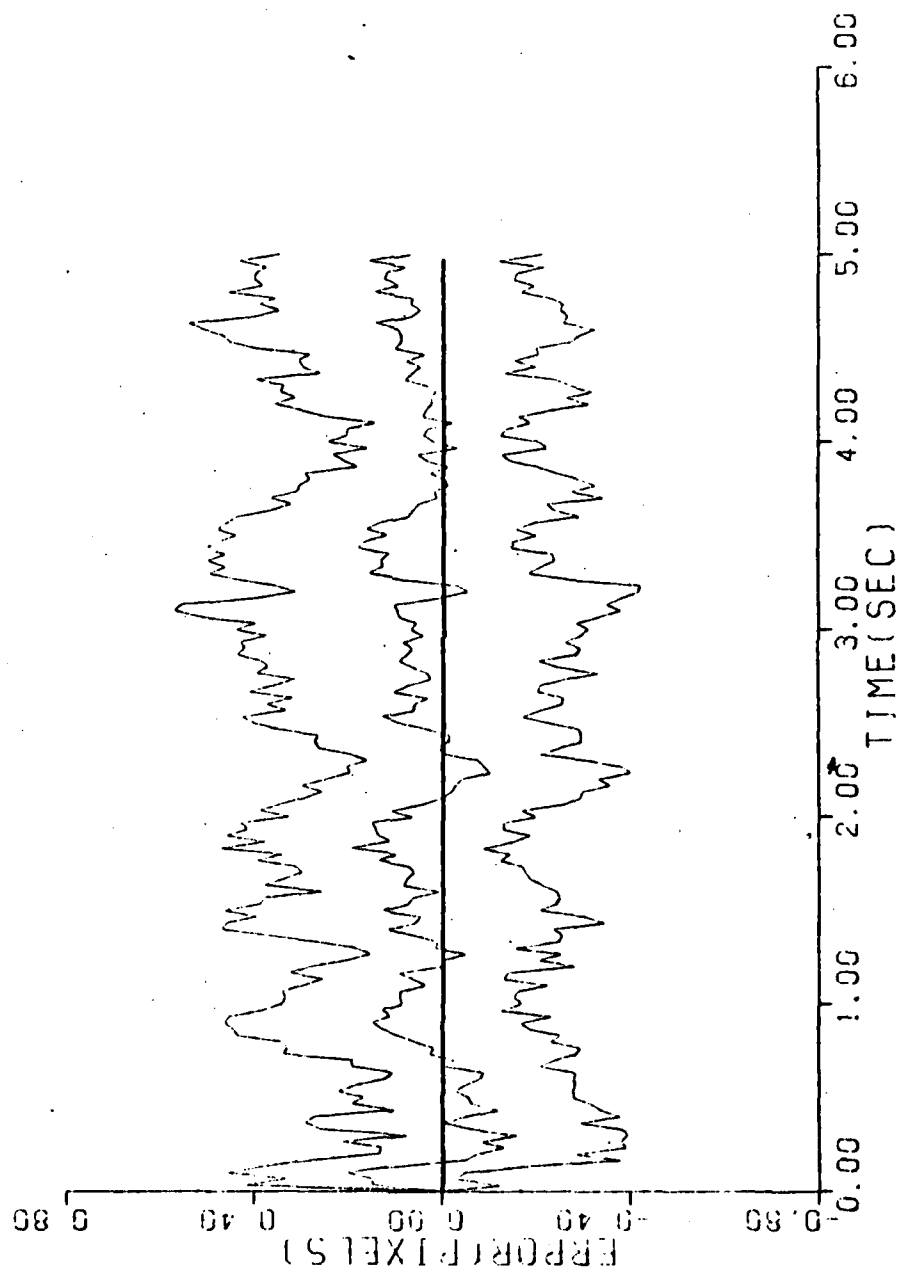


Figure C-4f. Case 4

ESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA

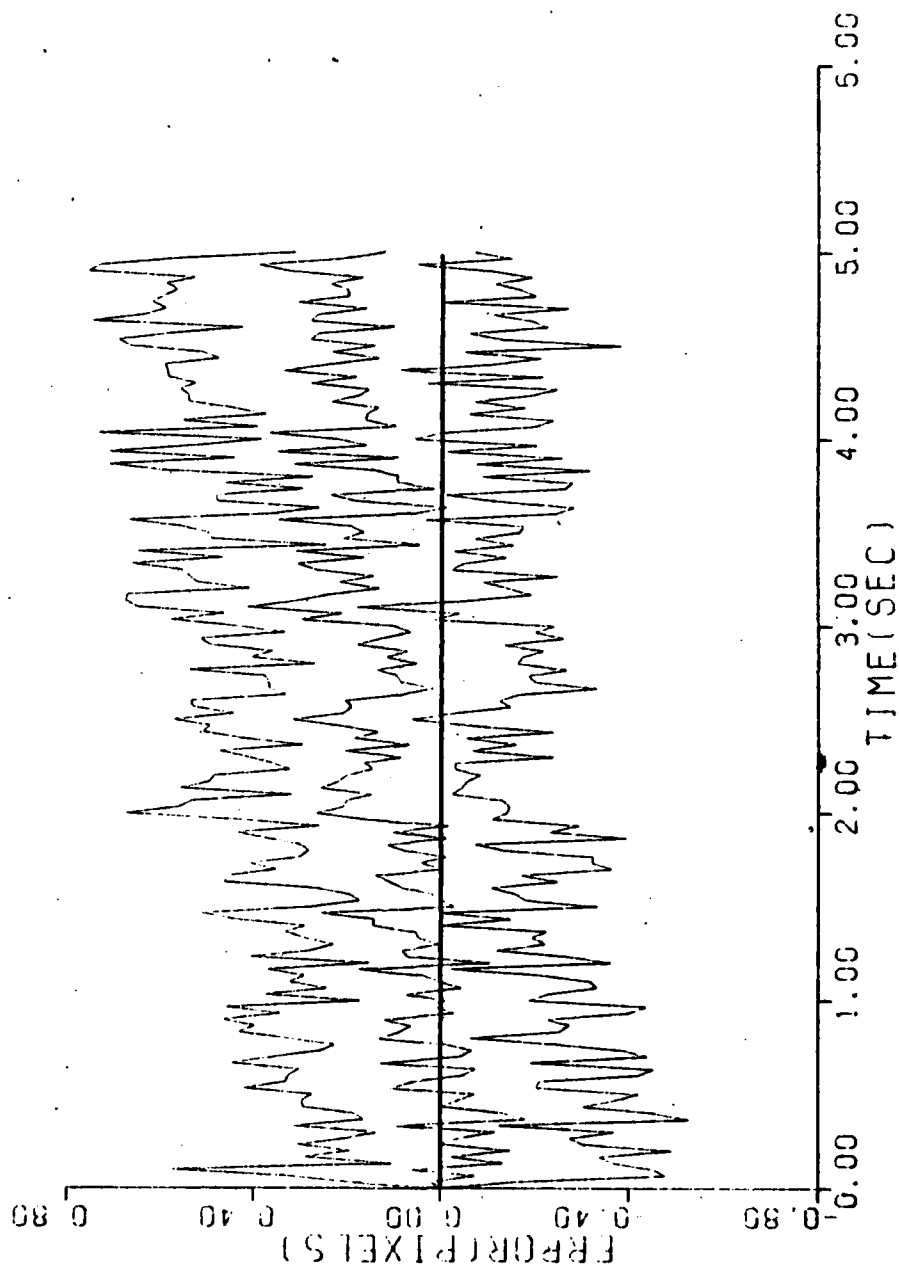


Figure C-4g. Case 4

ESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA

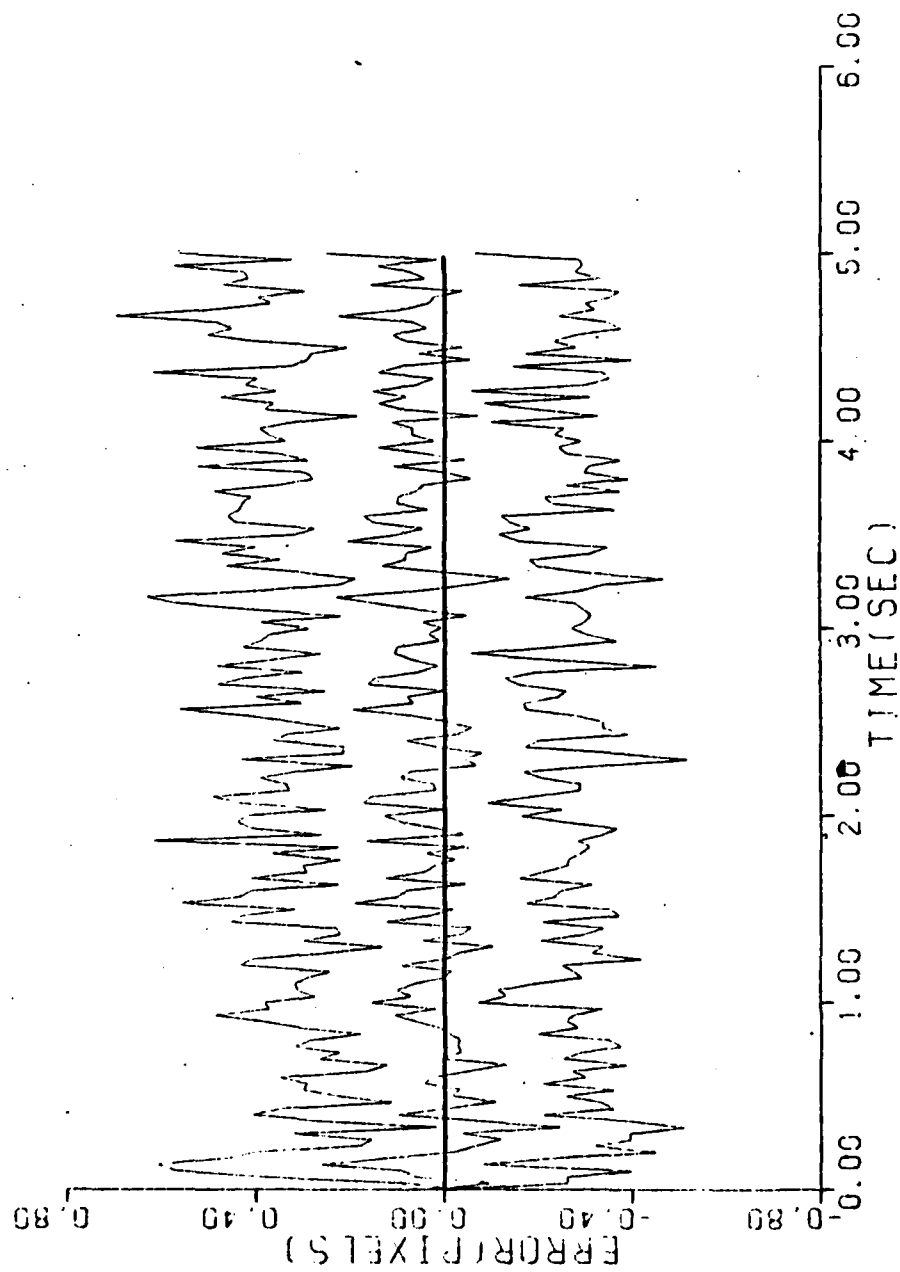


Figure C-4h. Case 4

ESTIMATED X-PLUS CENTROID POSITION (+/-) SIGMA

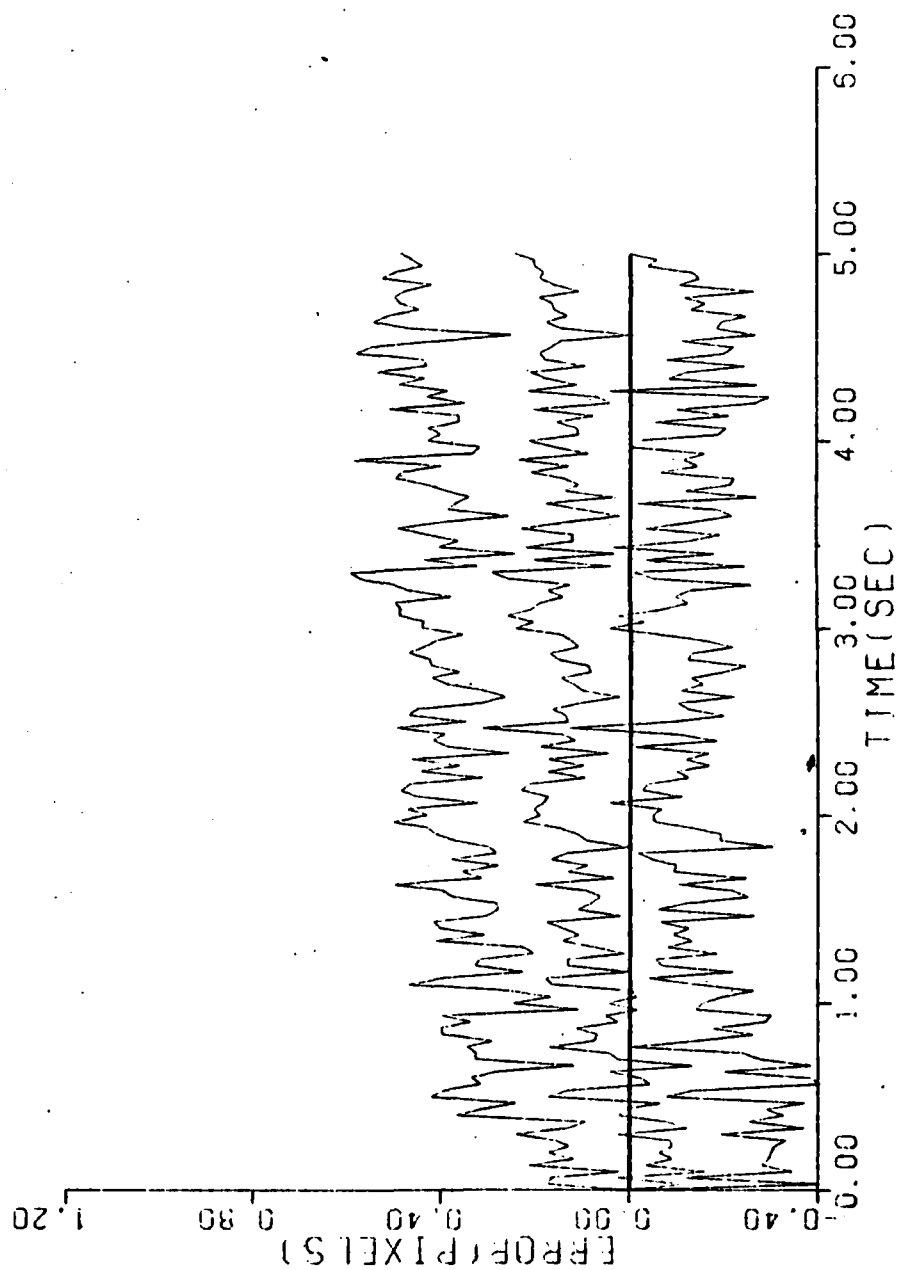


Figure C-4i. Case 4

ESTIMATED Y-PLUS CENTROID POSITION (+/-) SIGMA

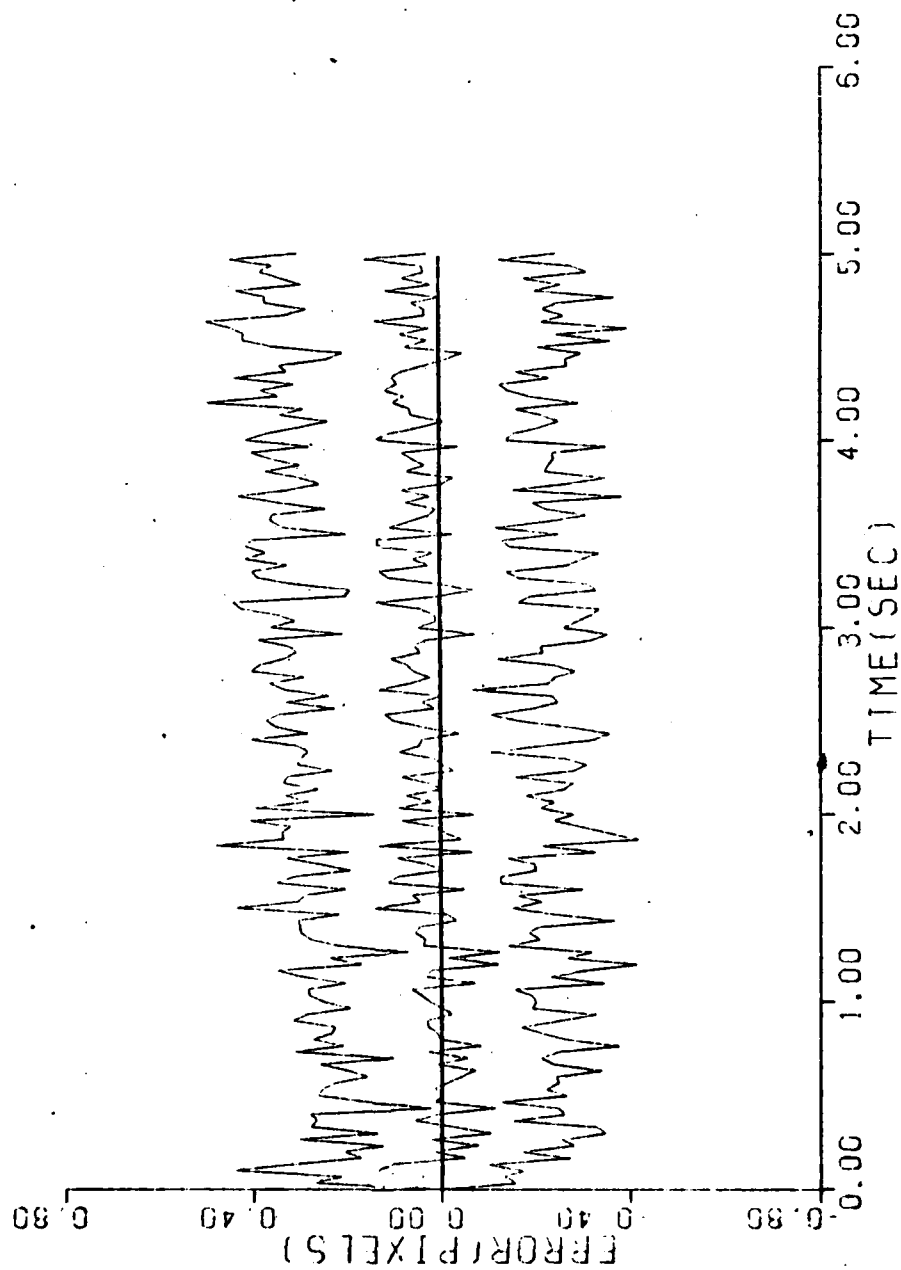


Figure C-4j. Case 4

The performance plots for this case were similar to the plots shown for case 3 and thus are omitted.

Figure C-5. Case 5

The performance plots for this case were similar to the plots shown for case 3, and thus are omitted.

Figure C-6. Case 6

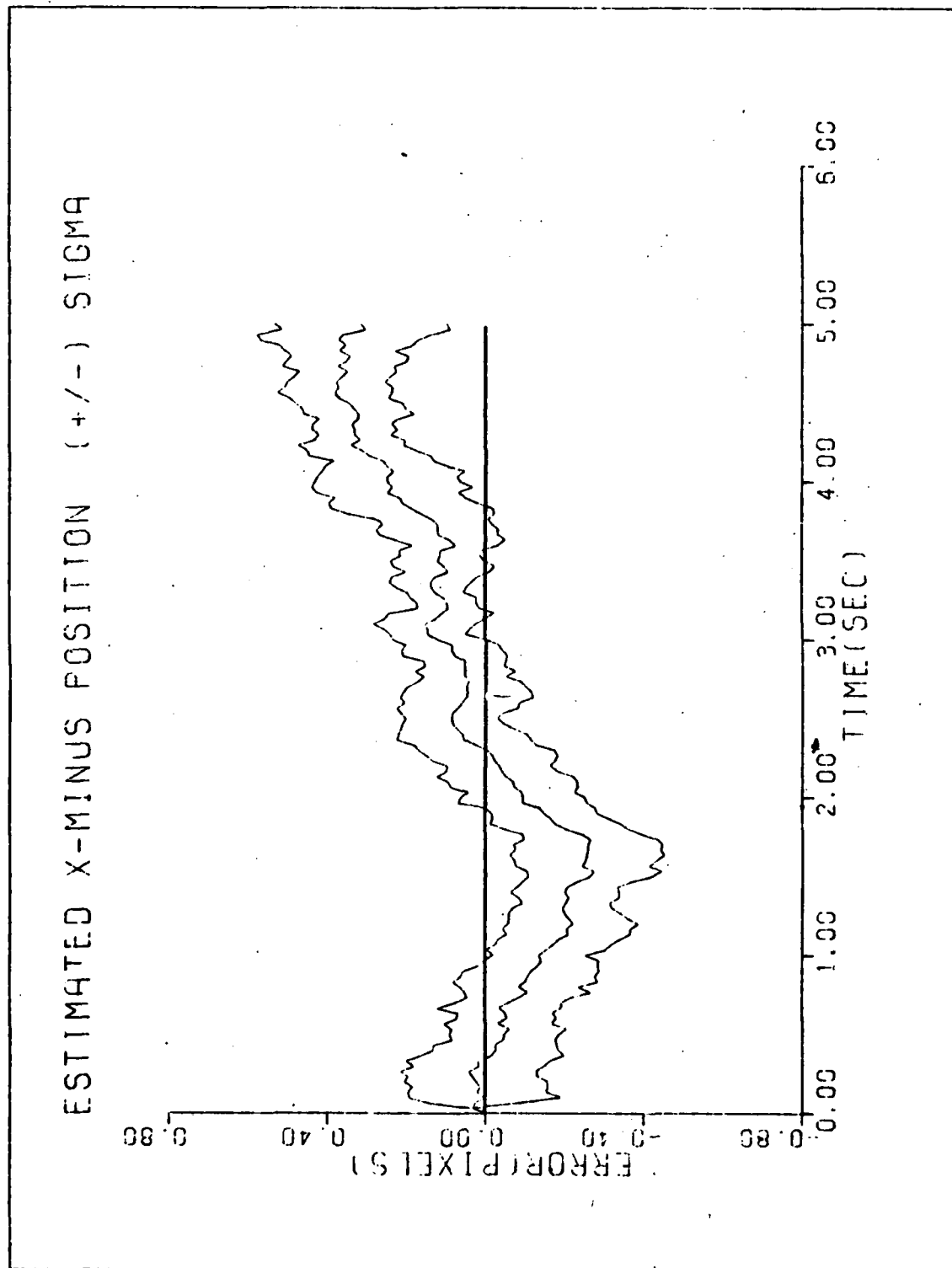


Figure C-7a. Case 7

ESTIMATED Y-MINUS POSITION (+/-) SIGMA

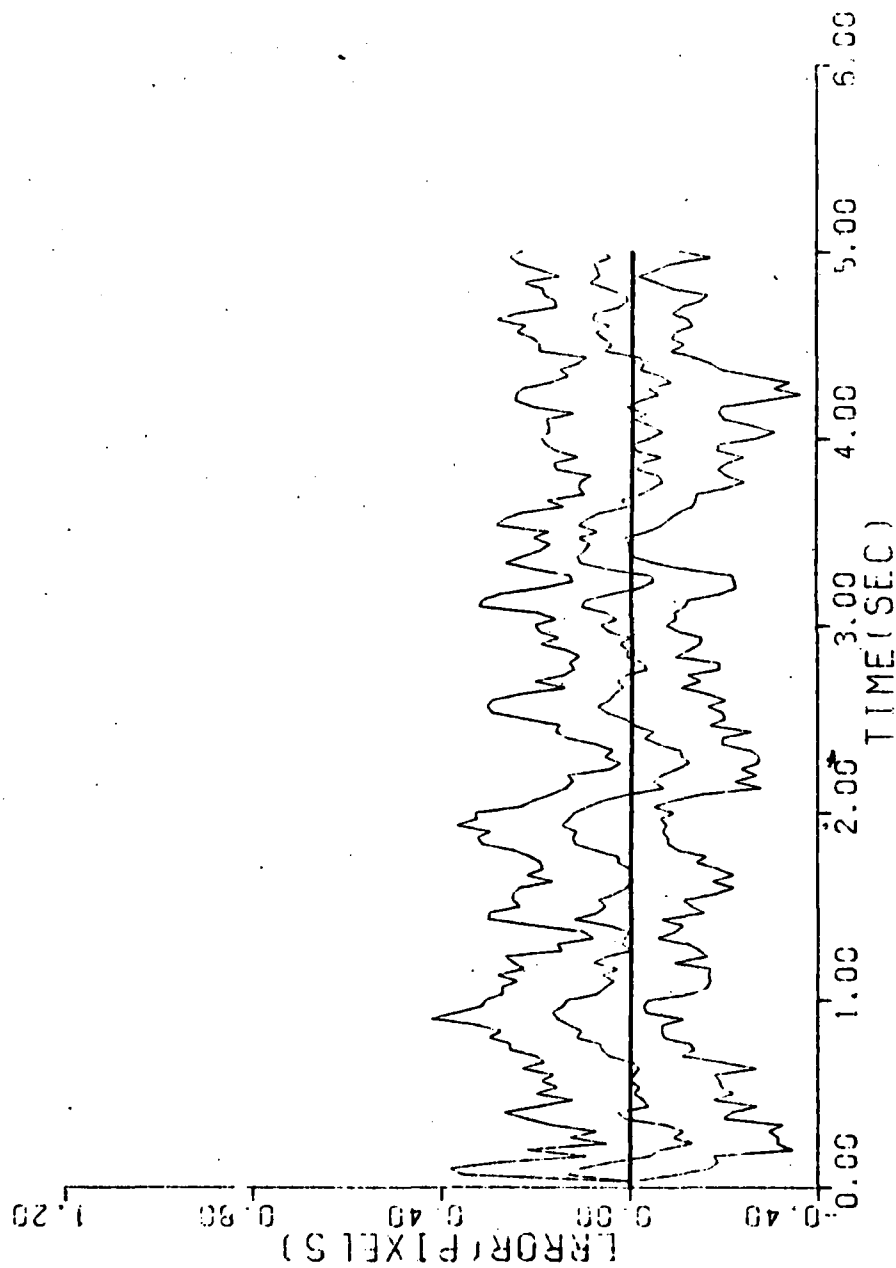


Figure C-7b. Case 7

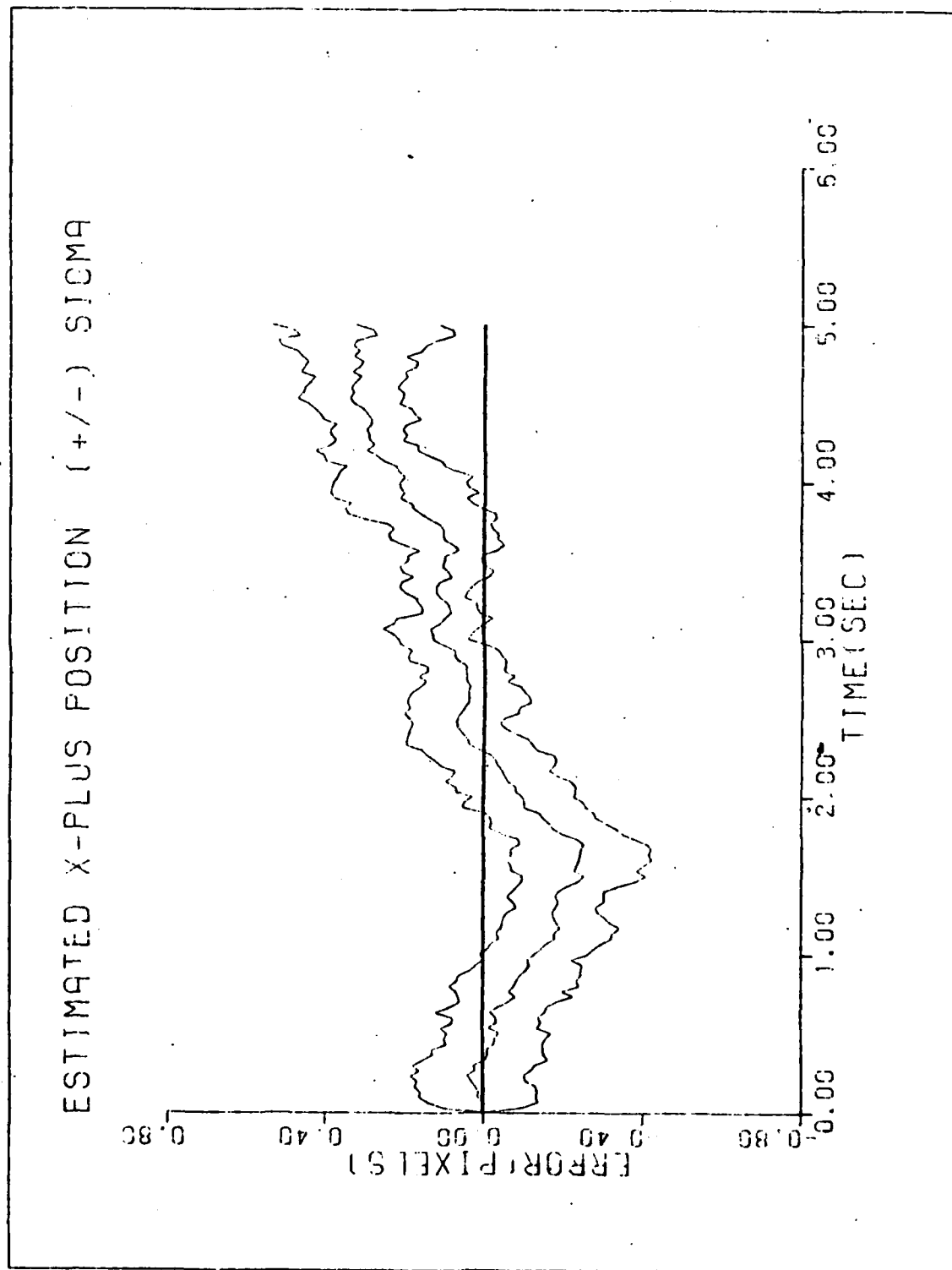


Figure C-7c. Case 7

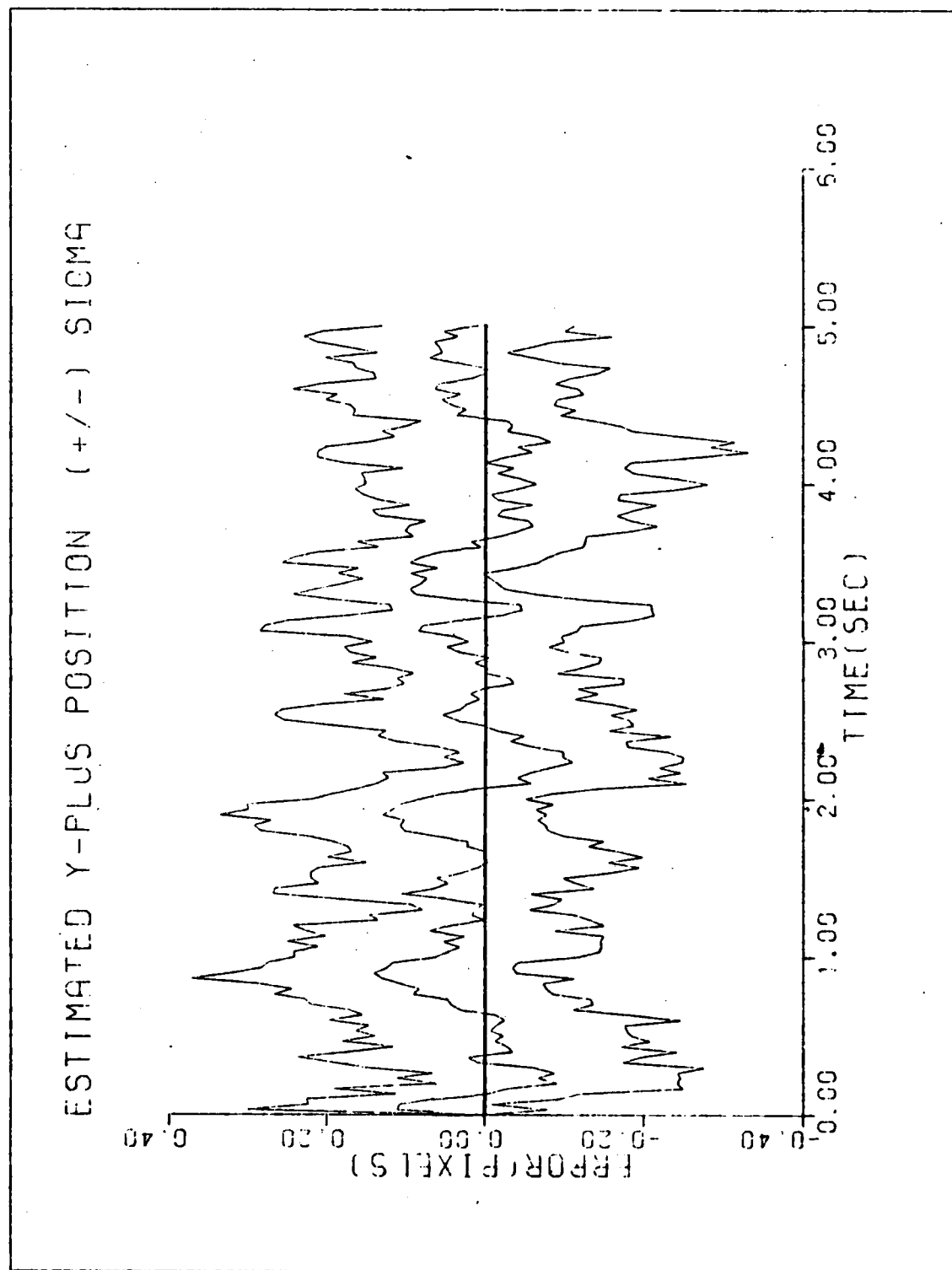


Figure C-7d. Case 7

ESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA

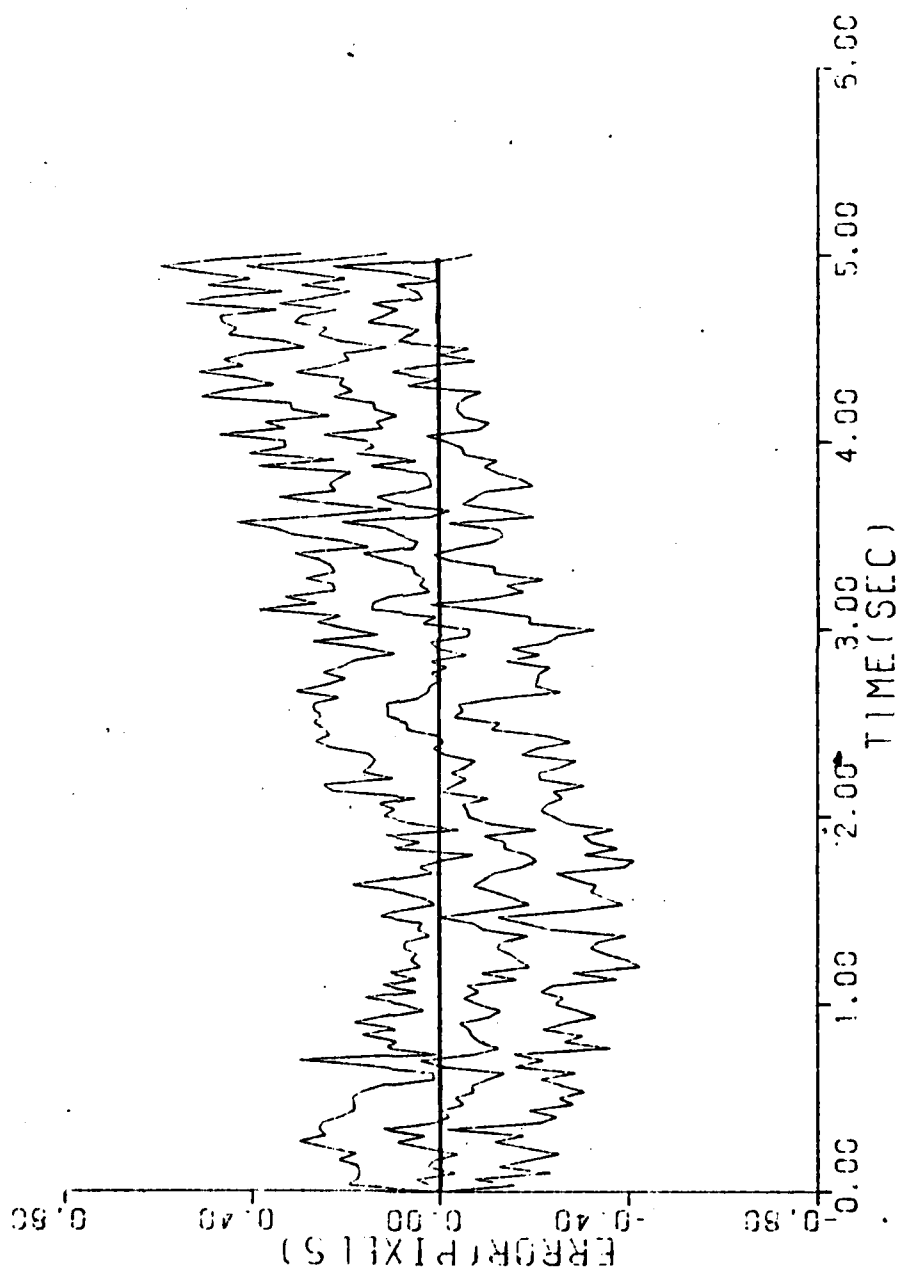


Figure C-7e. Case 7

ESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA

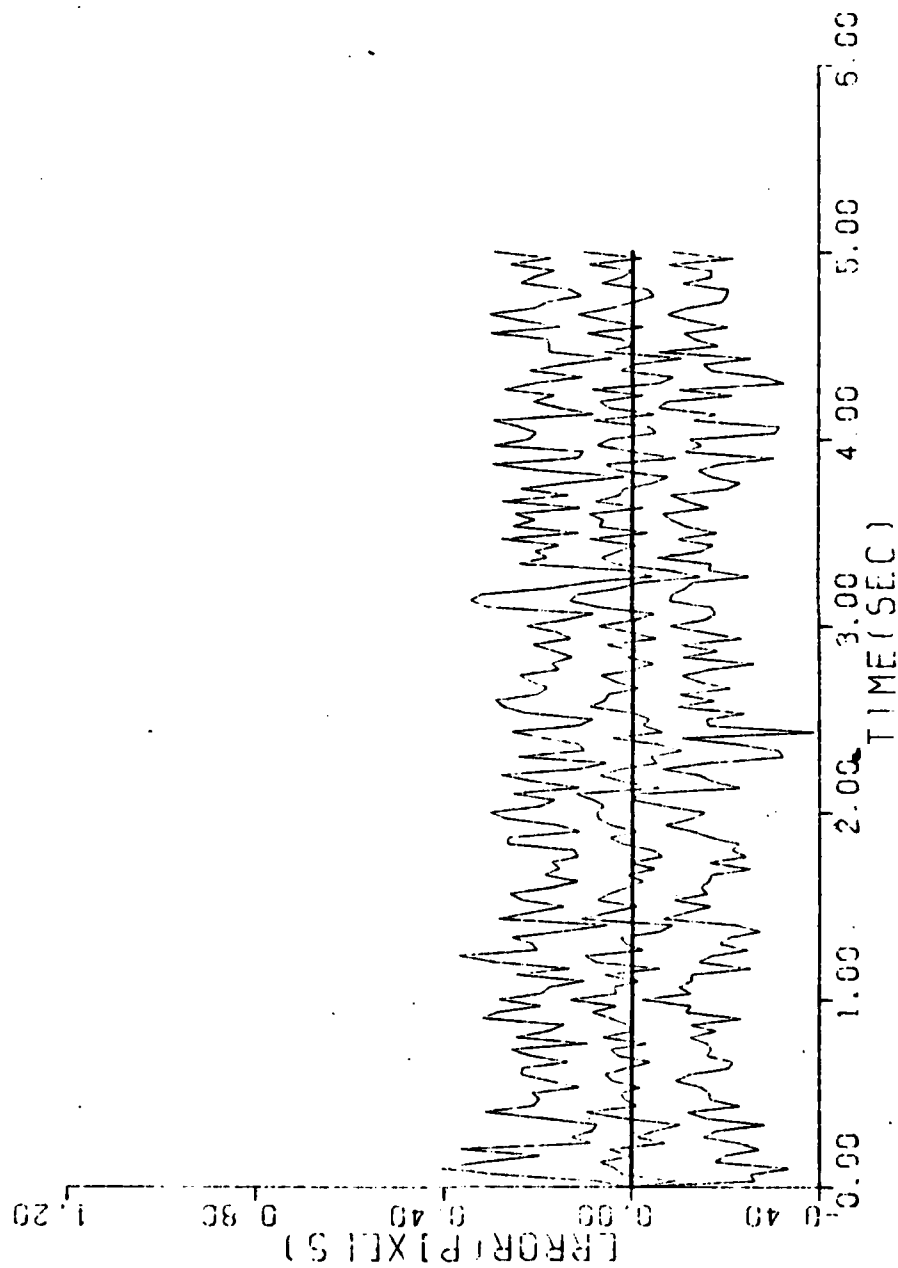


Figure C-7f. Case 7

ESTIMATED X-PLUS CENTROID POSITION (+/-) SIGMA

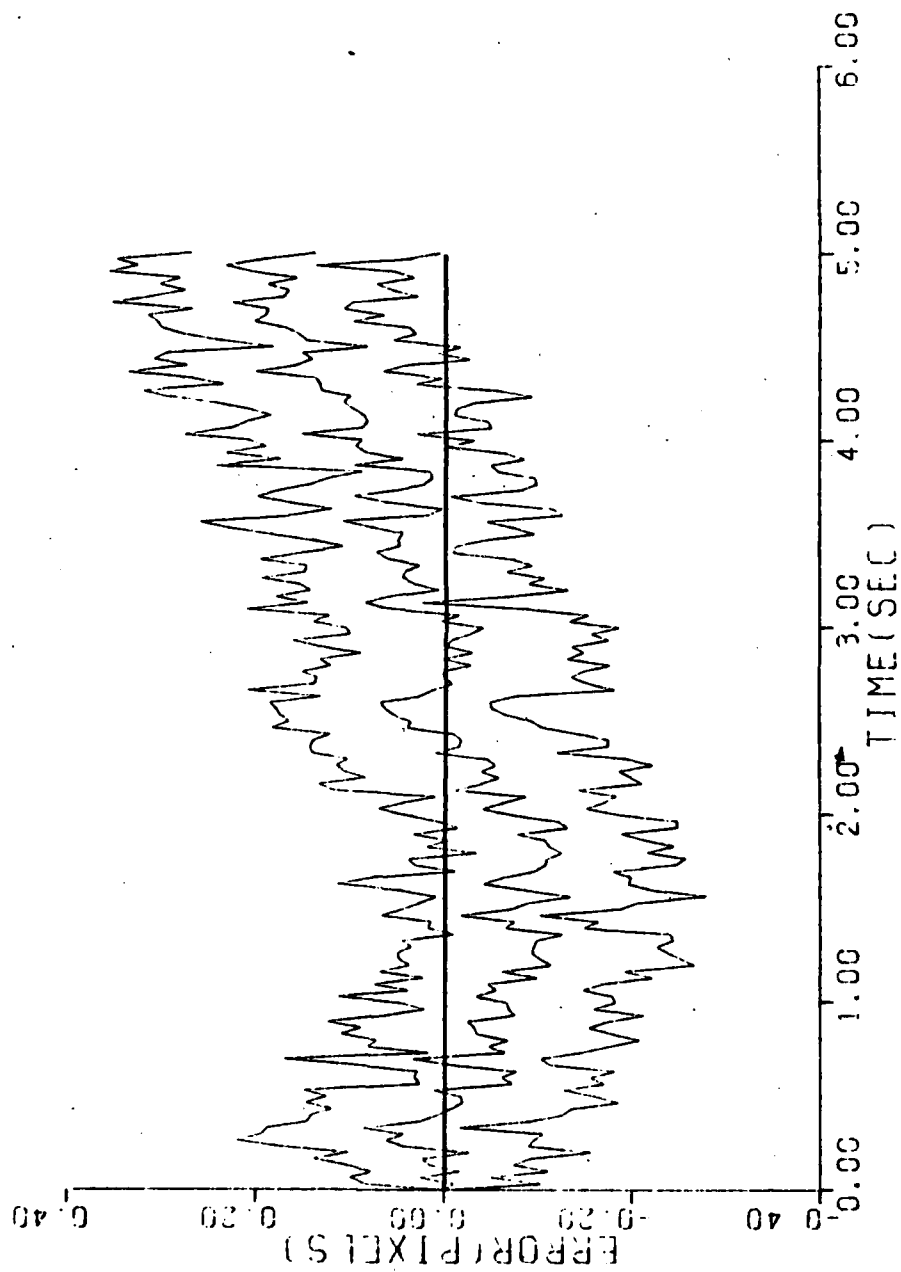


Figure C-7g. Case 7

ESTIMATED Y-PLUS CENTROID POSITION (+/-) SIGMA

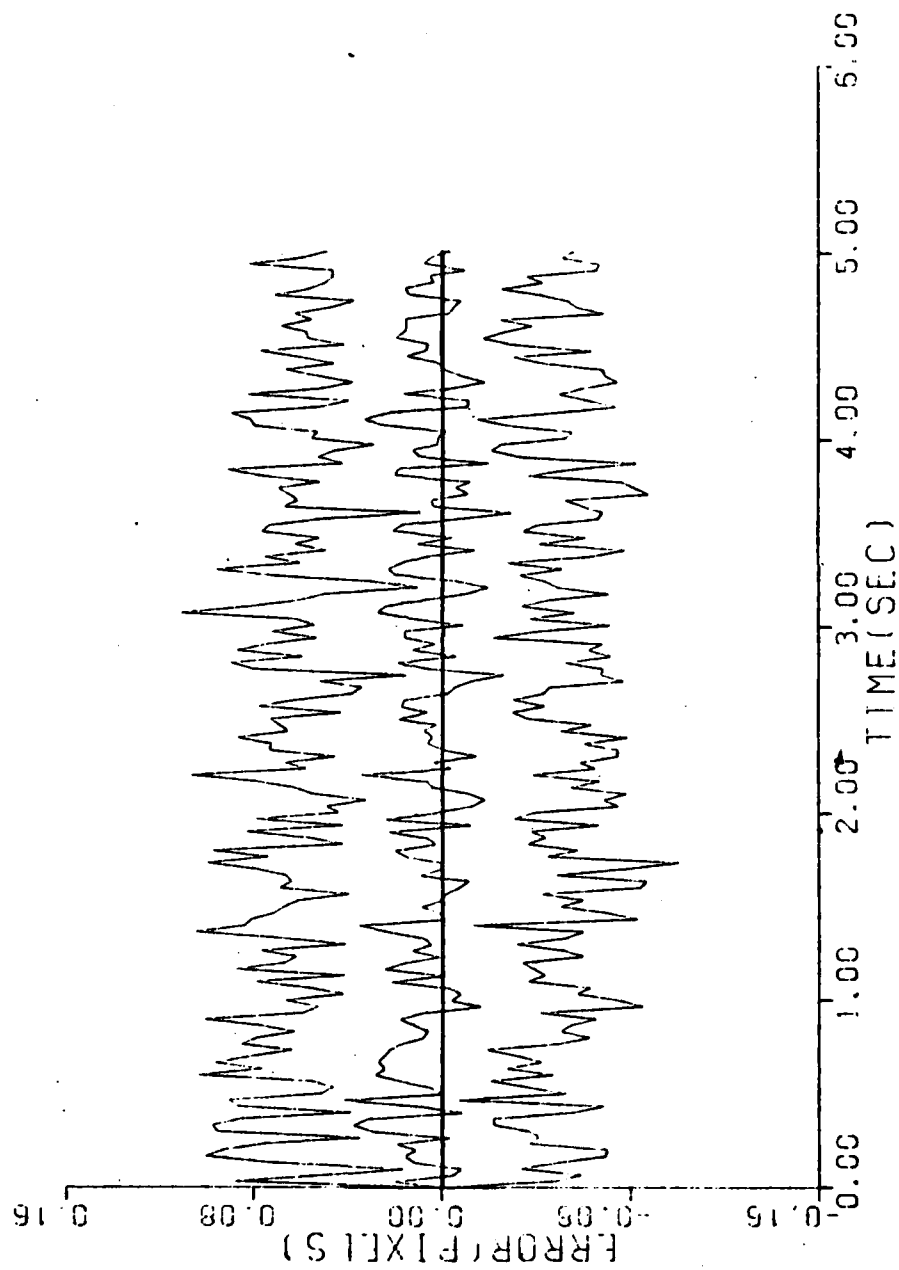


Figure C-7h. Case 7

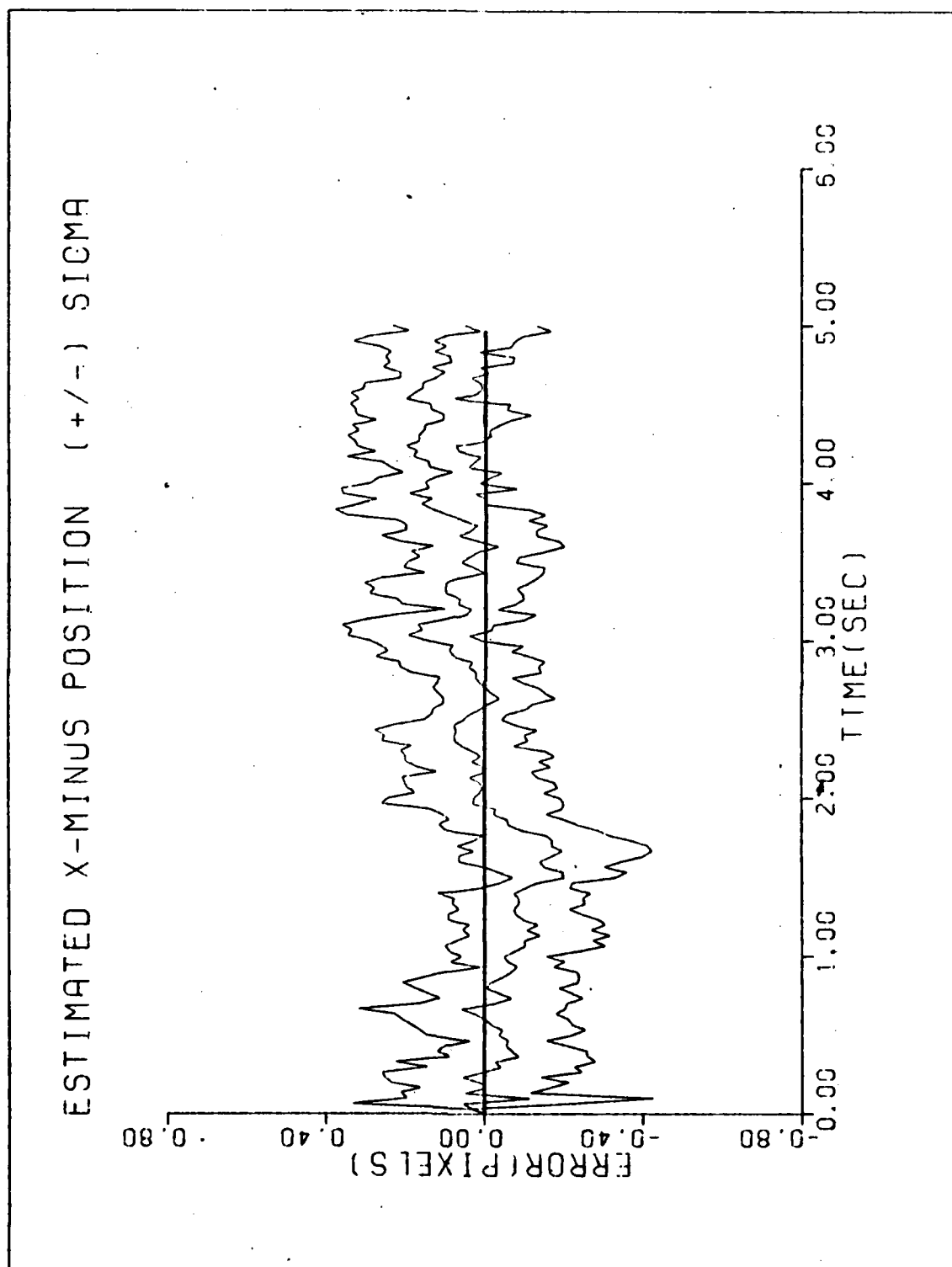


Figure C-8a. Case 8

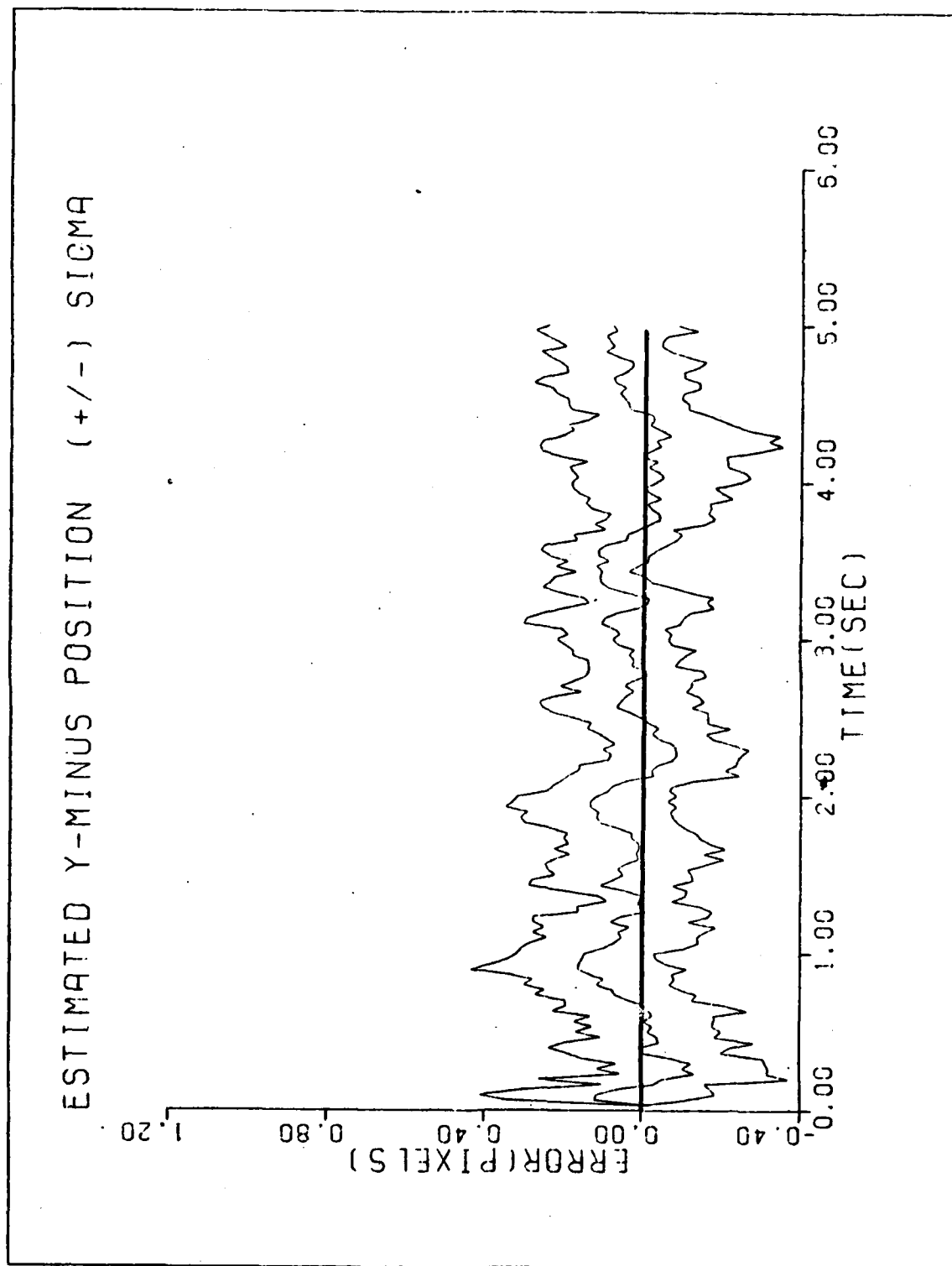


Figure C-8b. Case 8

ESTIMATED X-PLUS POSITION (+/-) SIGMA

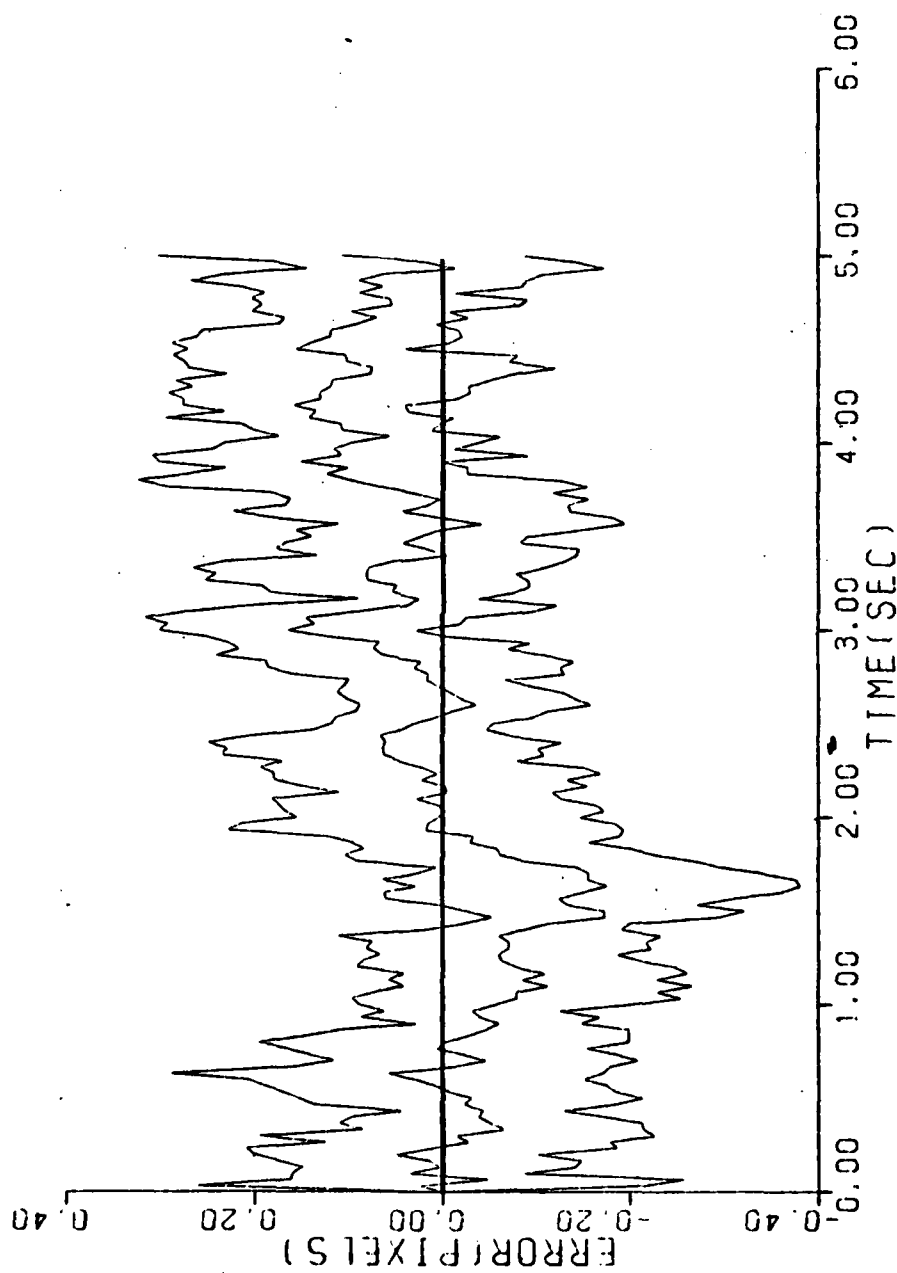


Figure C-8c. Case 8

ESTIMATED Y-PLUS POSITION (+/-) SIGMA

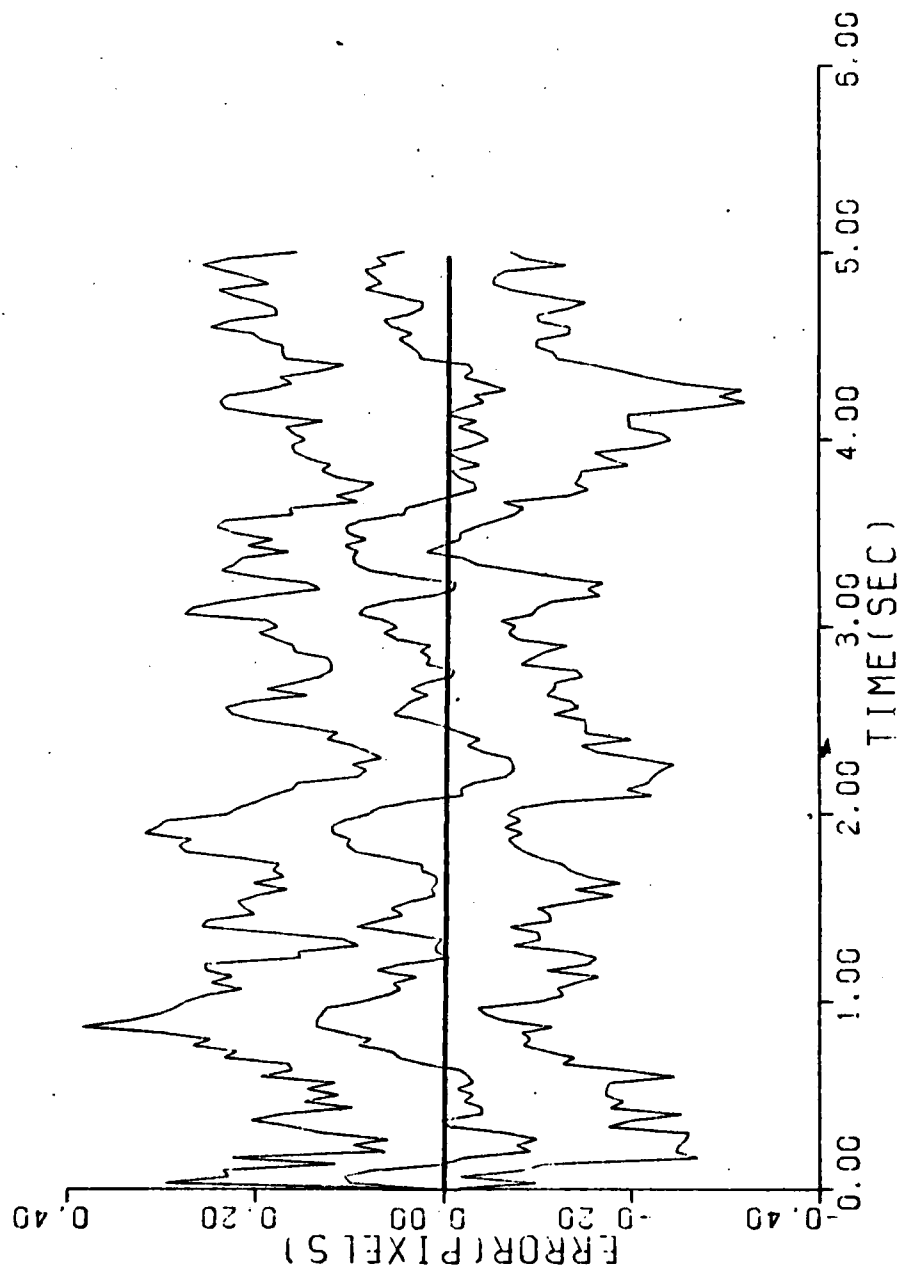


Figure C-8d. Case 8

ESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA

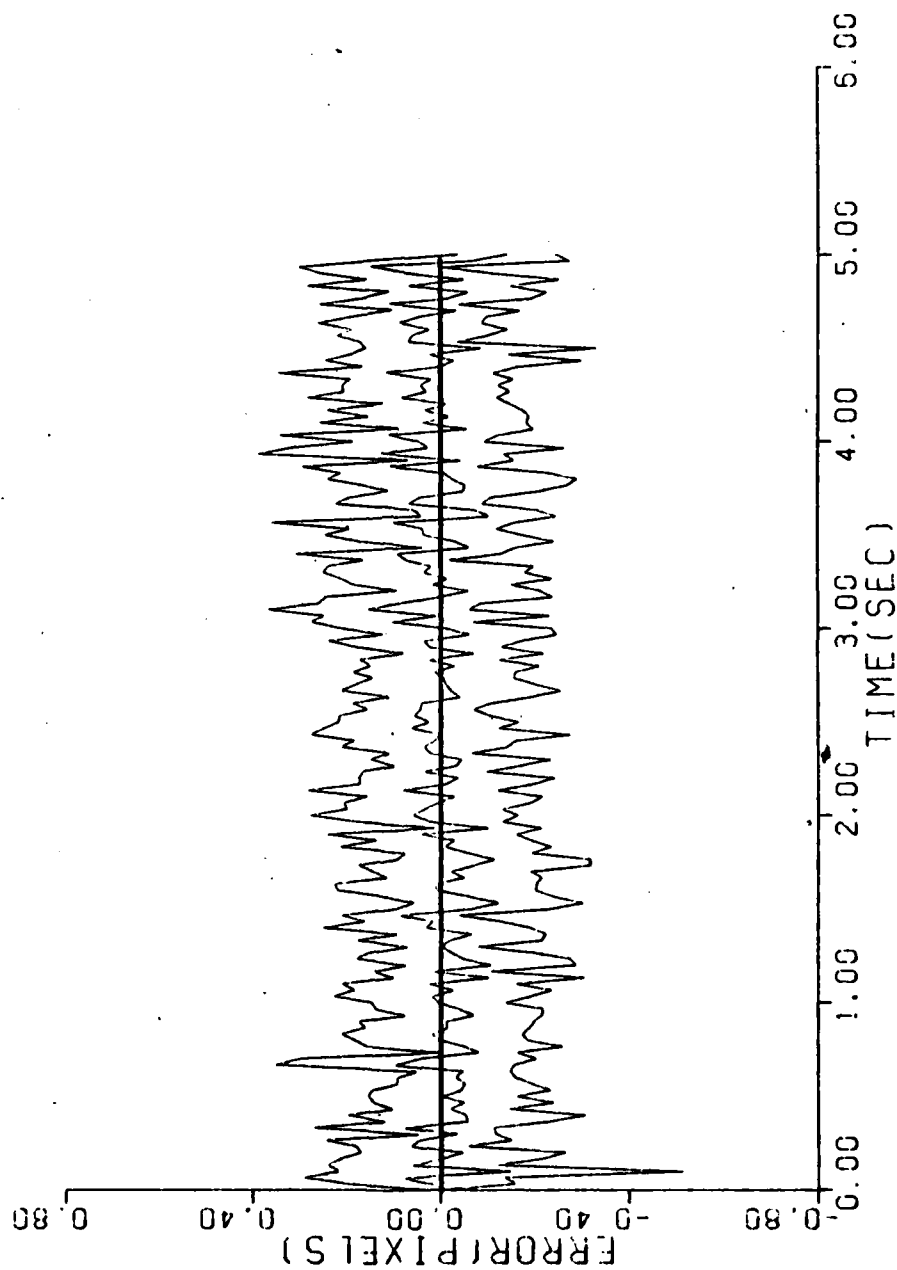


Figure C-8e. Case 8

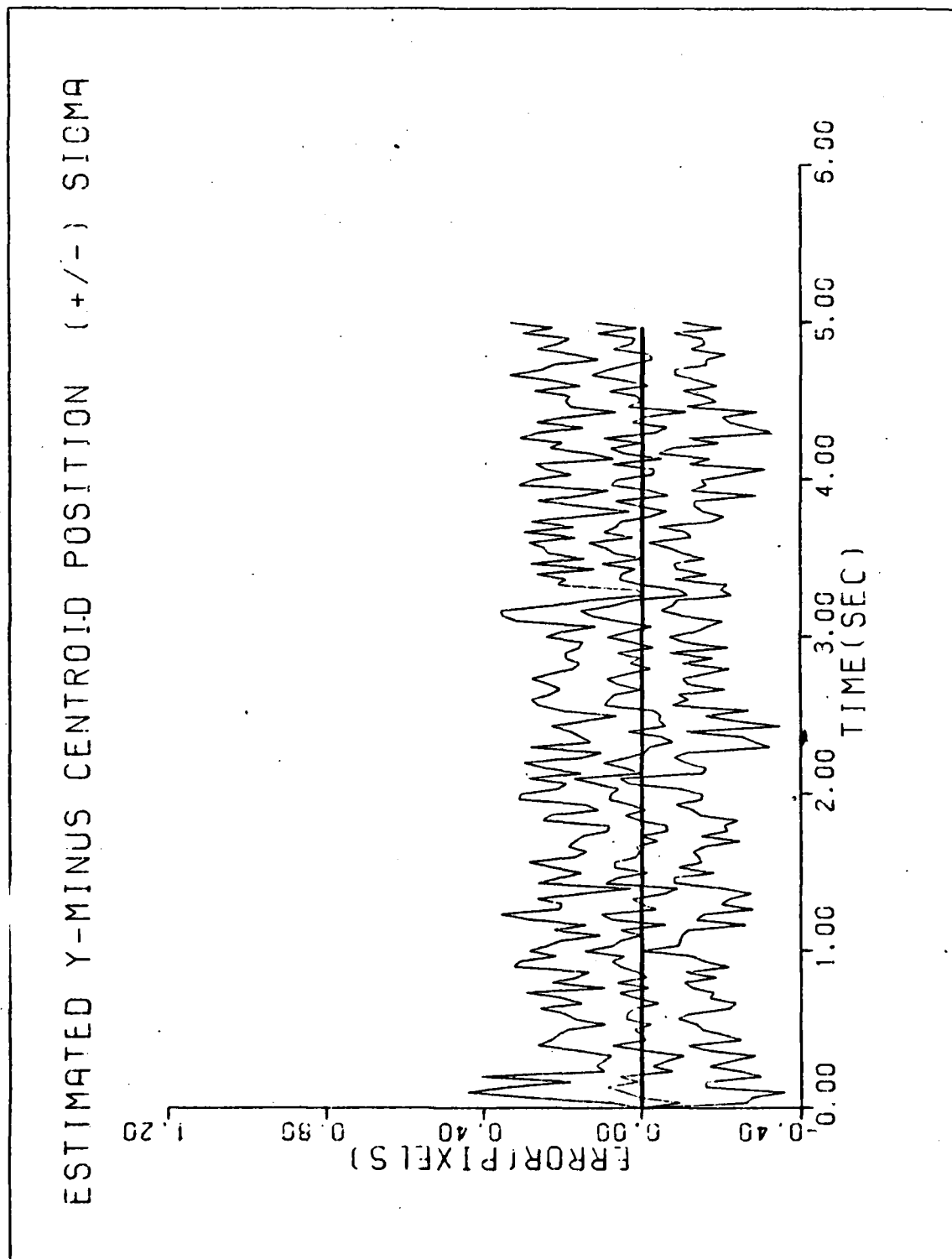


Figure C-8f. Case 8

The performance plots for this case were similar to the plots shown for case 8, and thus are omitted.

Figure C-9. Case 9

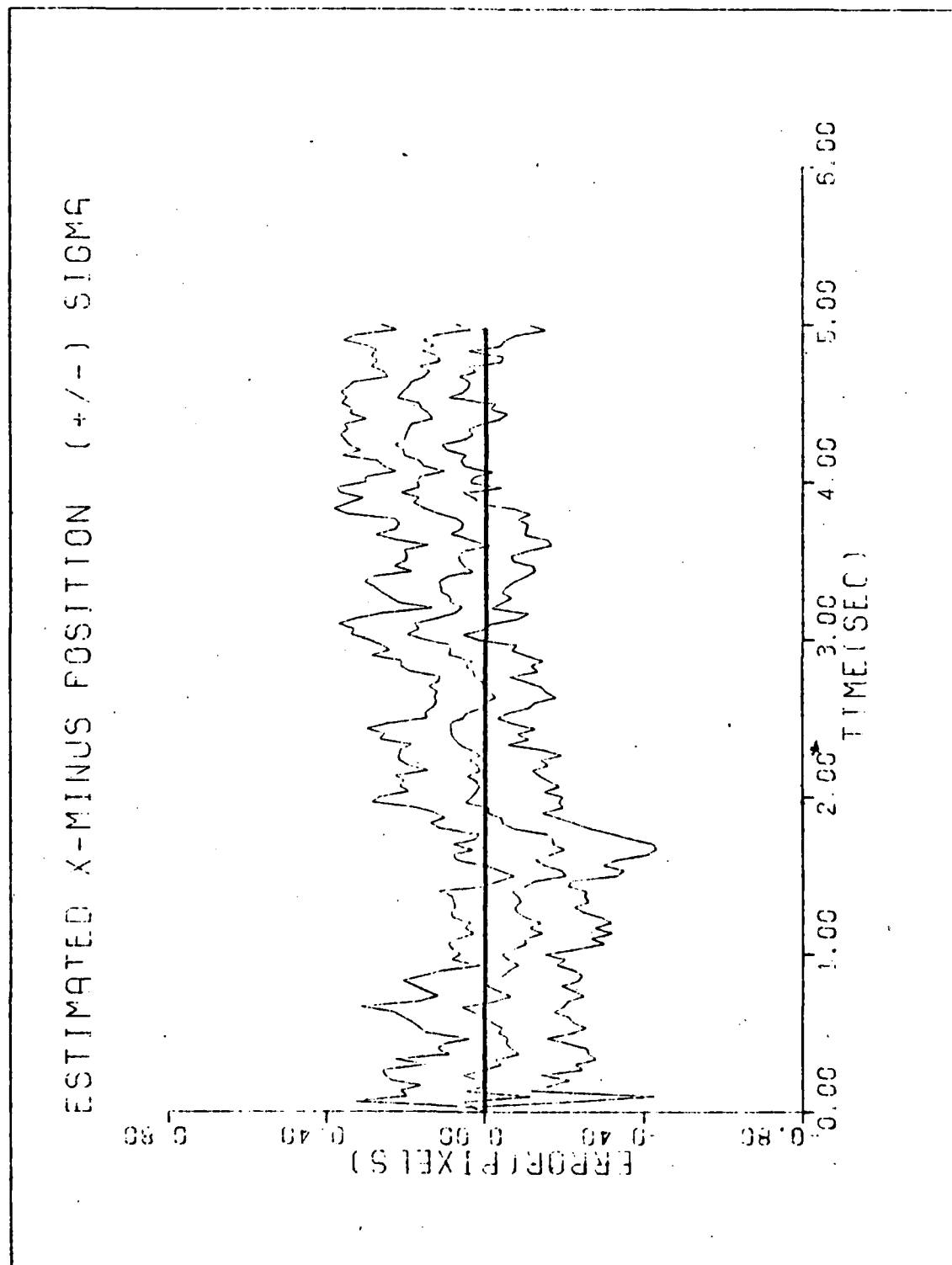


Figure C-10a. Case 10

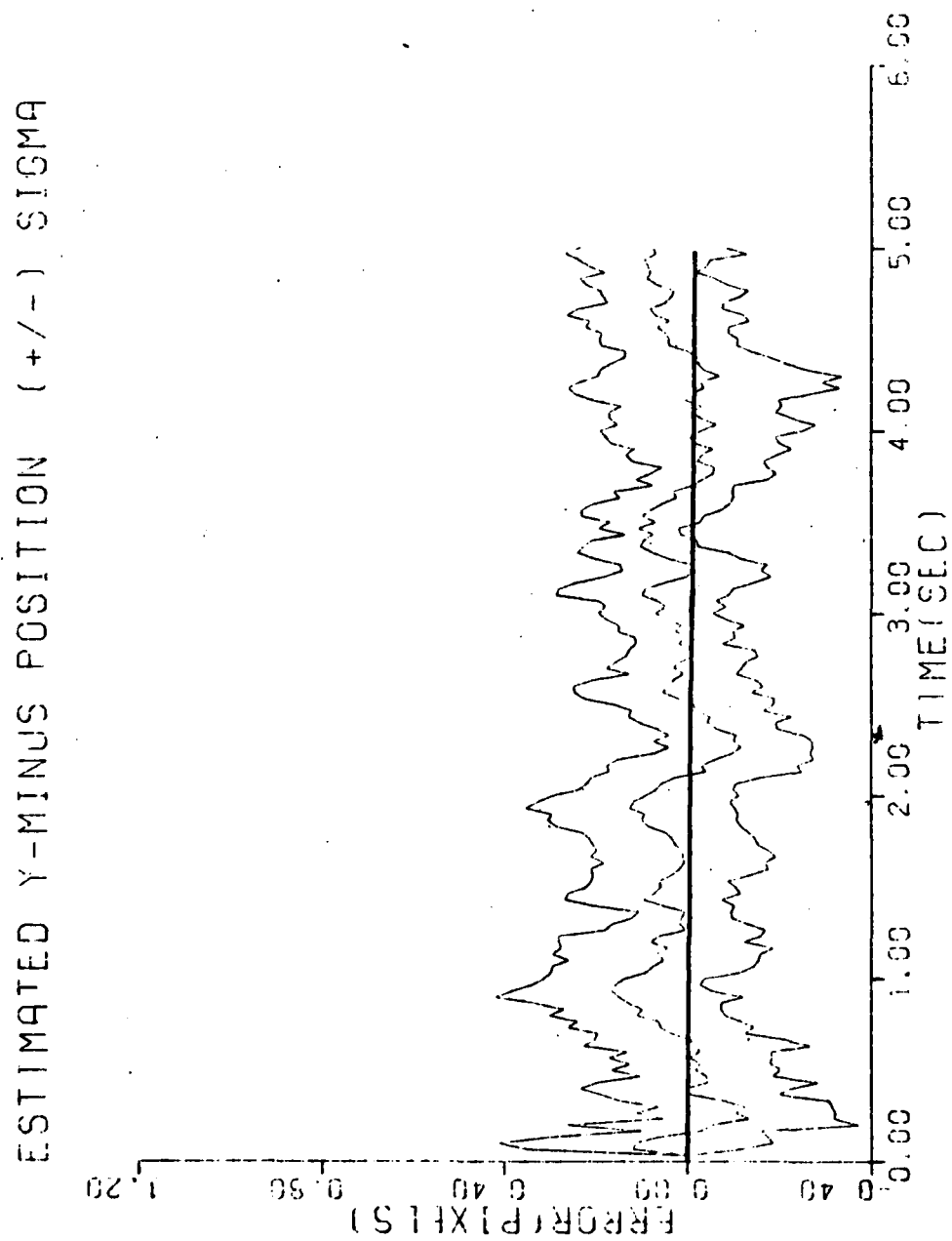


Figure C-10b. Case 10

ESTIMATED X-PLUS POSITION (+/-) SIGMA

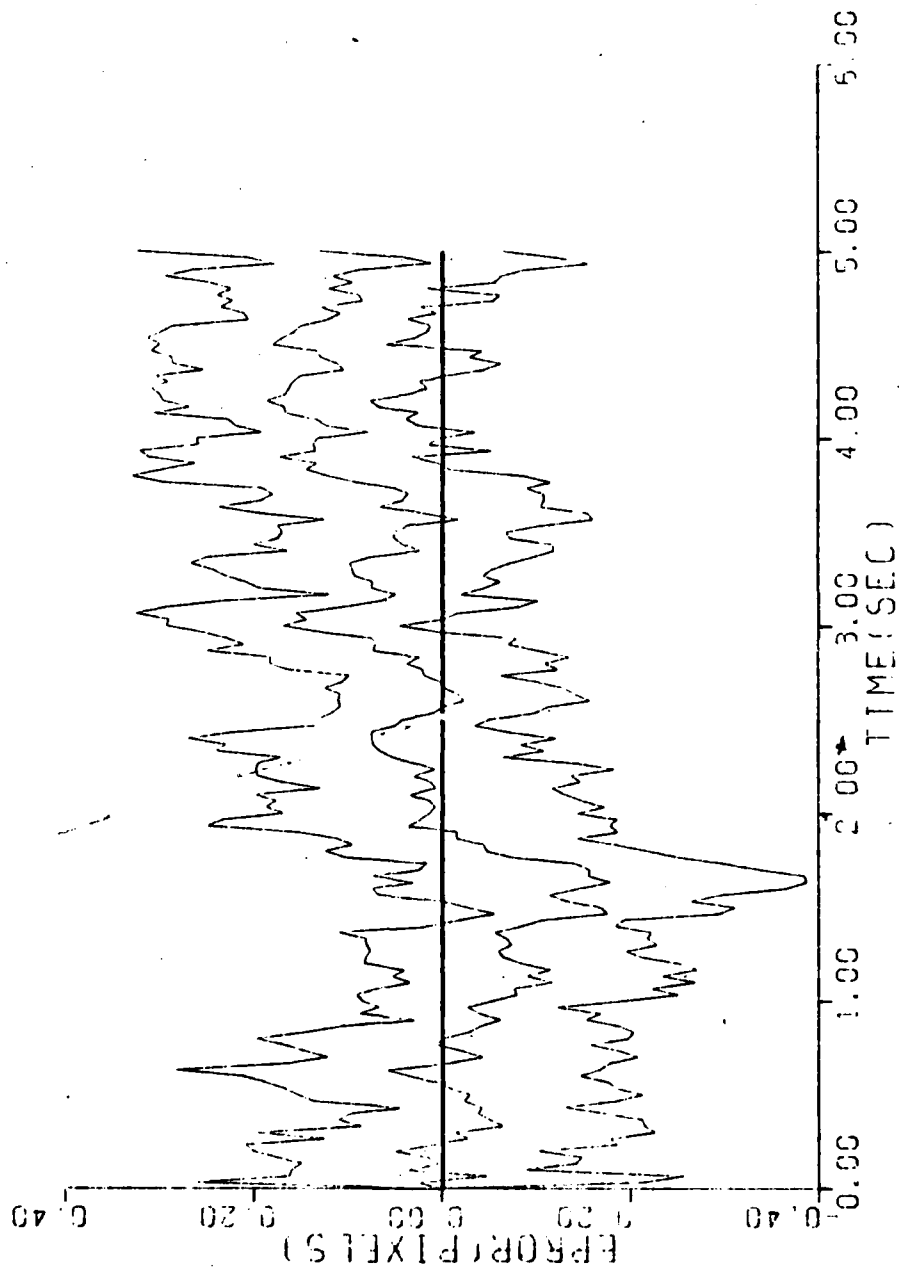


Figure C-10c. Case 10

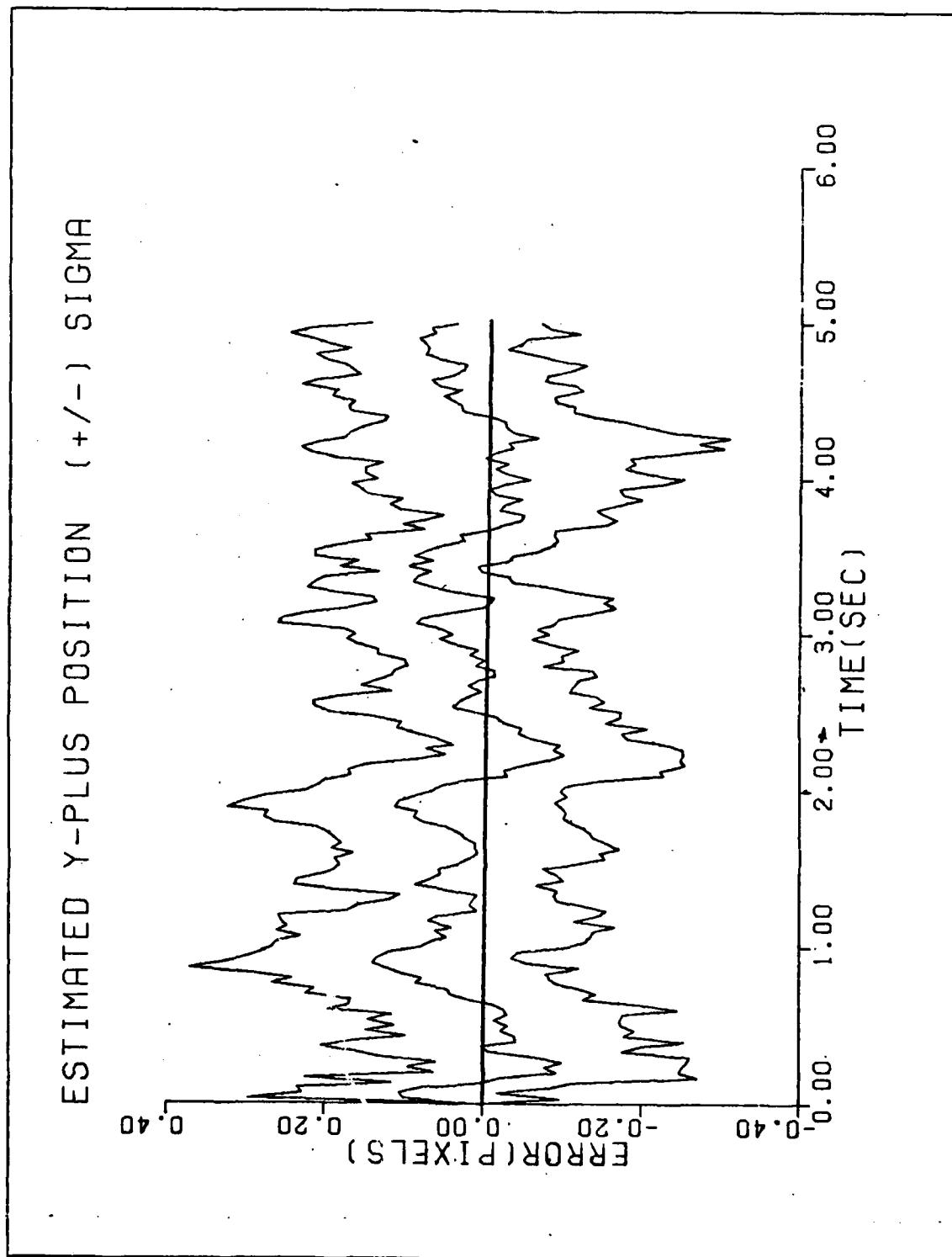


Figure C-10d. Case 10

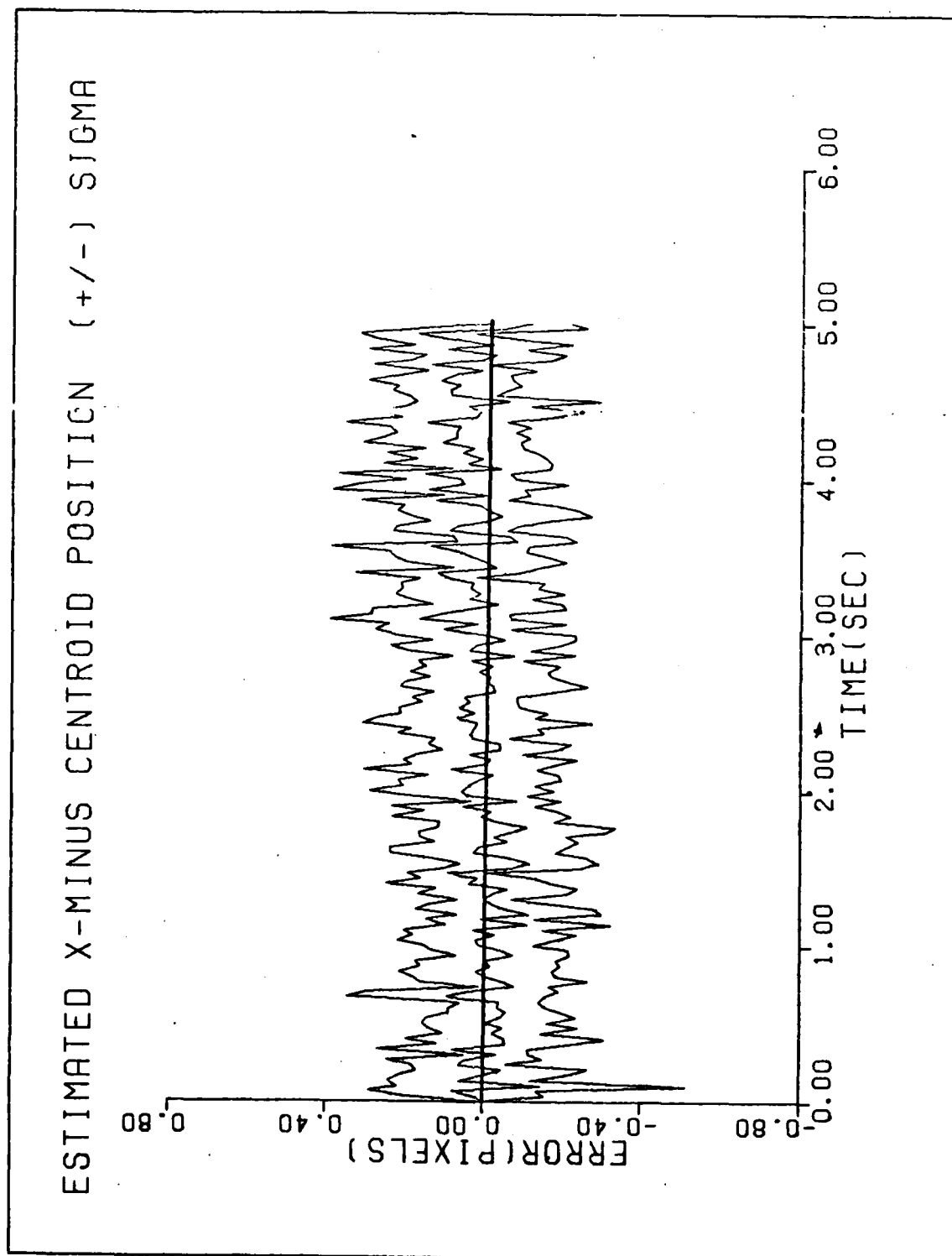


Figure C-10e. Case 10

ESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA

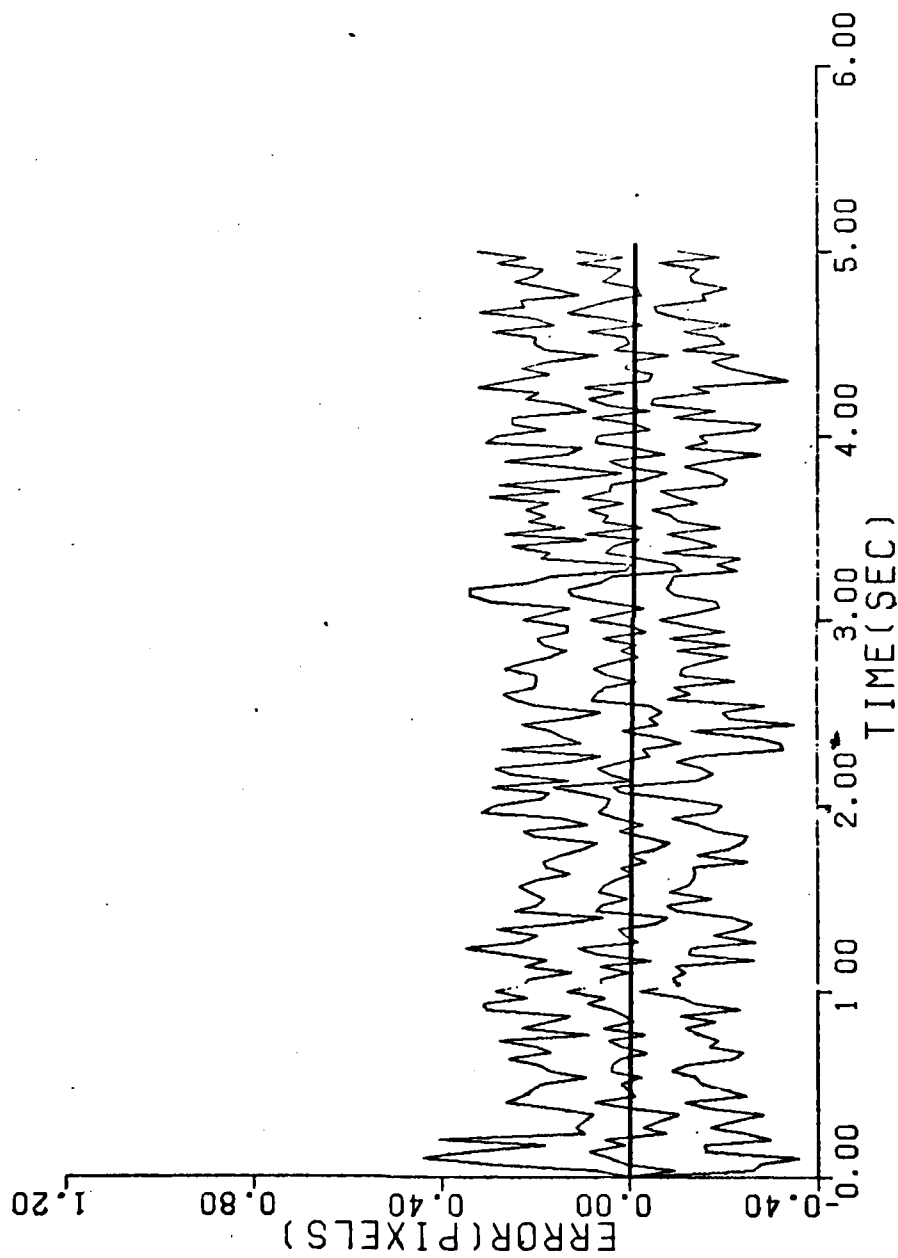


Figure C-10f. Case 10

The performance plots for this case were similar to the plots shown for case 10, and thus are omitted.

Figure C-11. Case 11

ESTIMATED X-MINUS POSITION (---) SIGMA

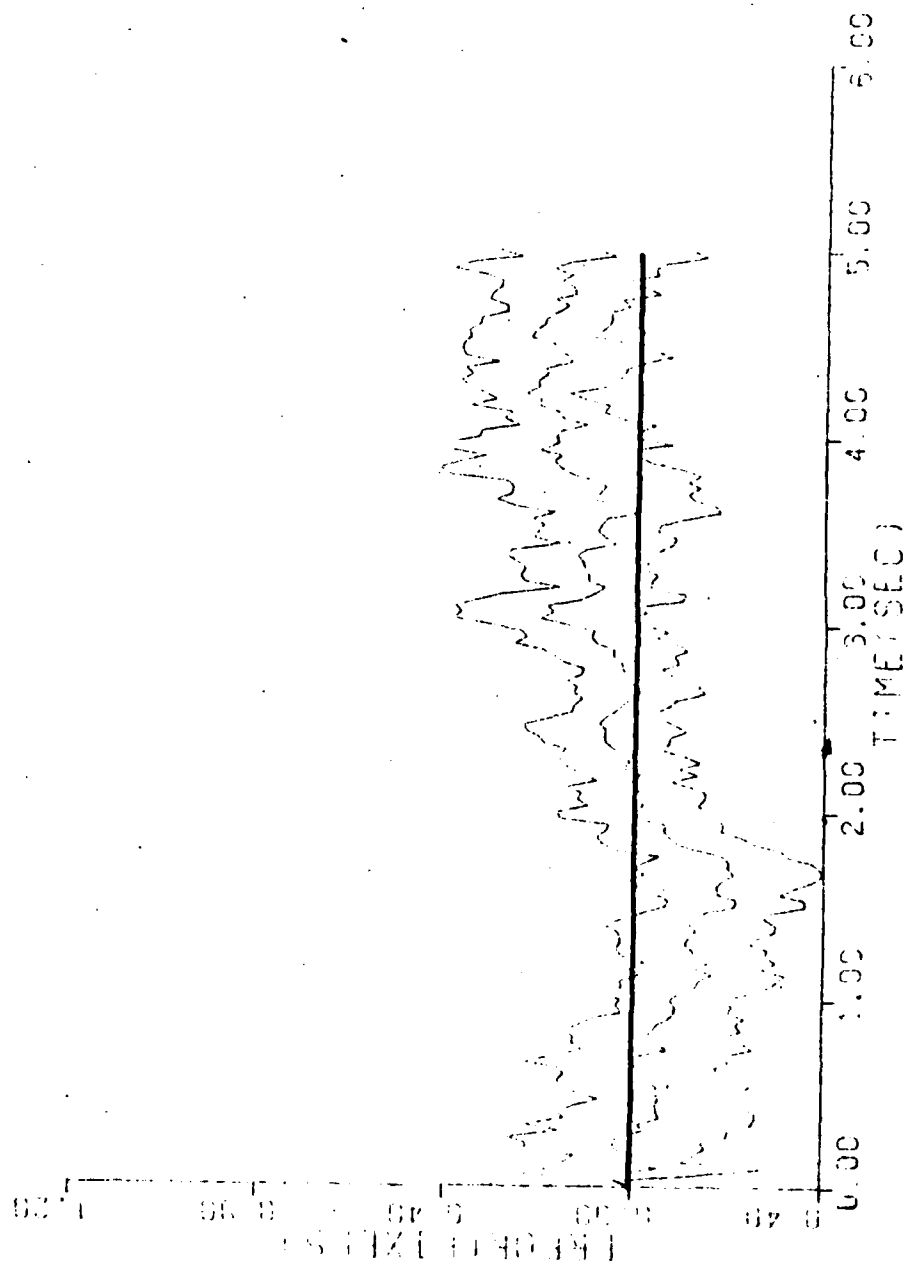


Figure C-12a. Case 12

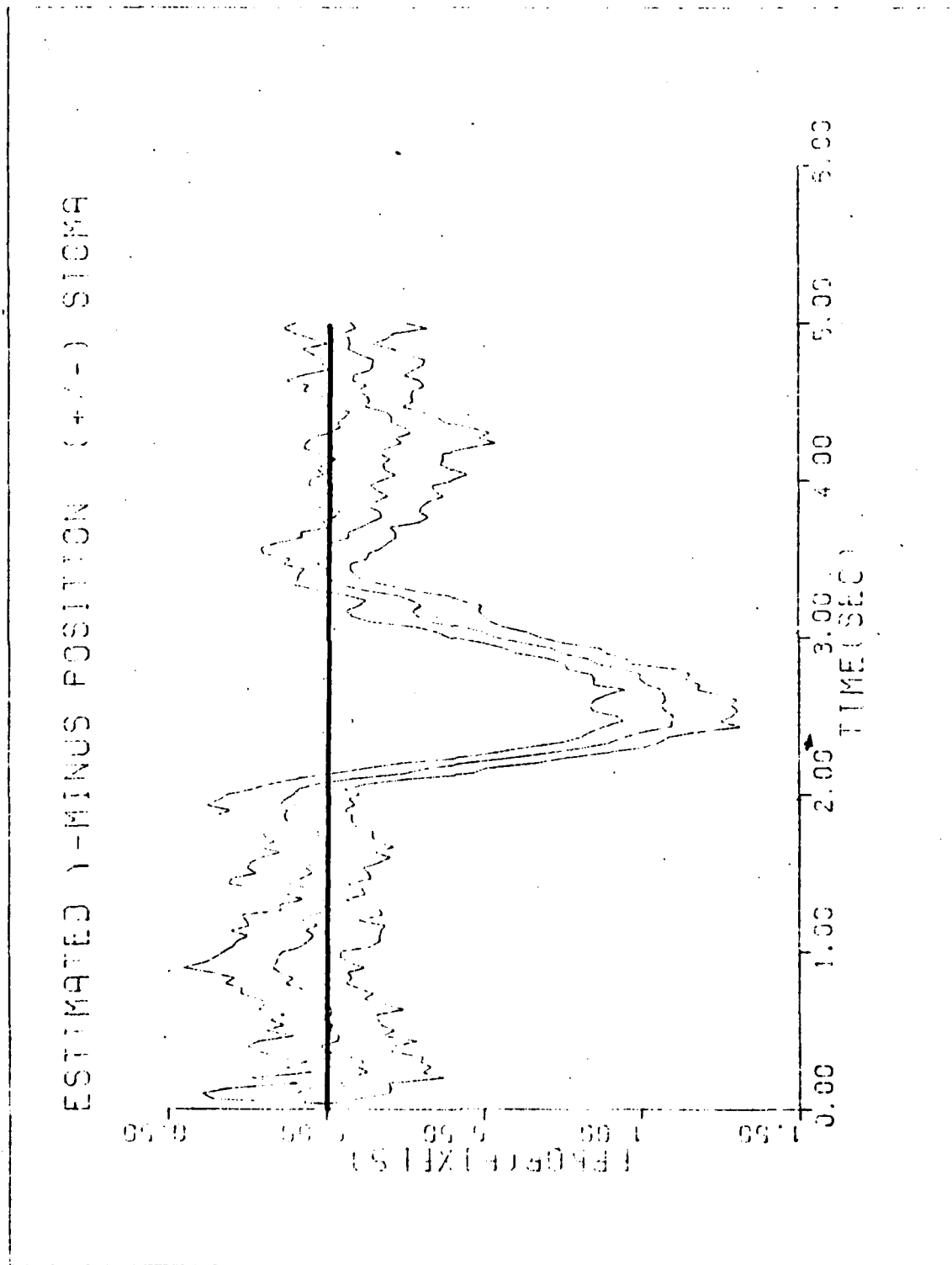


Figure C-12b. Case 12

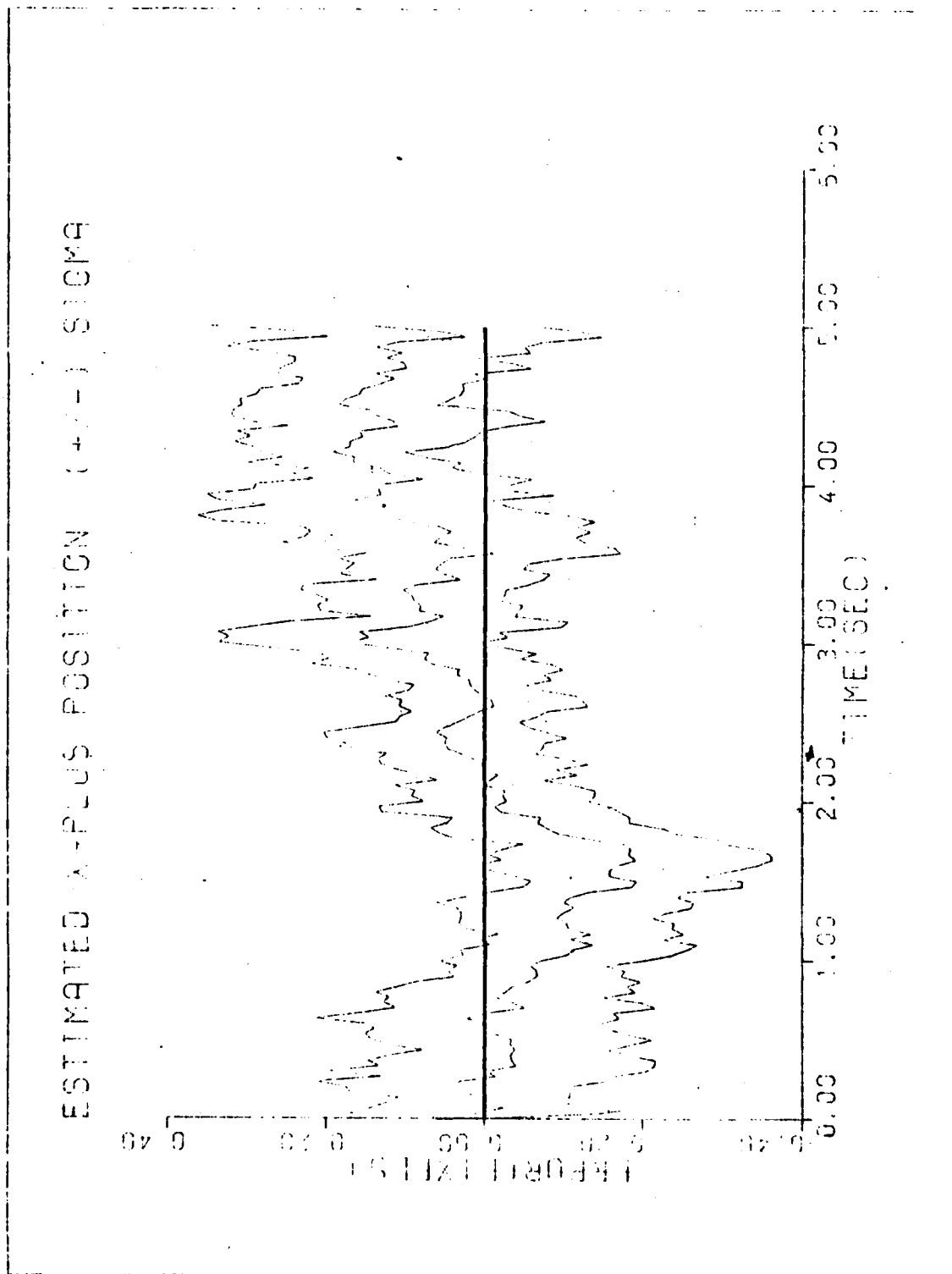


Figure C-12c. Case 12

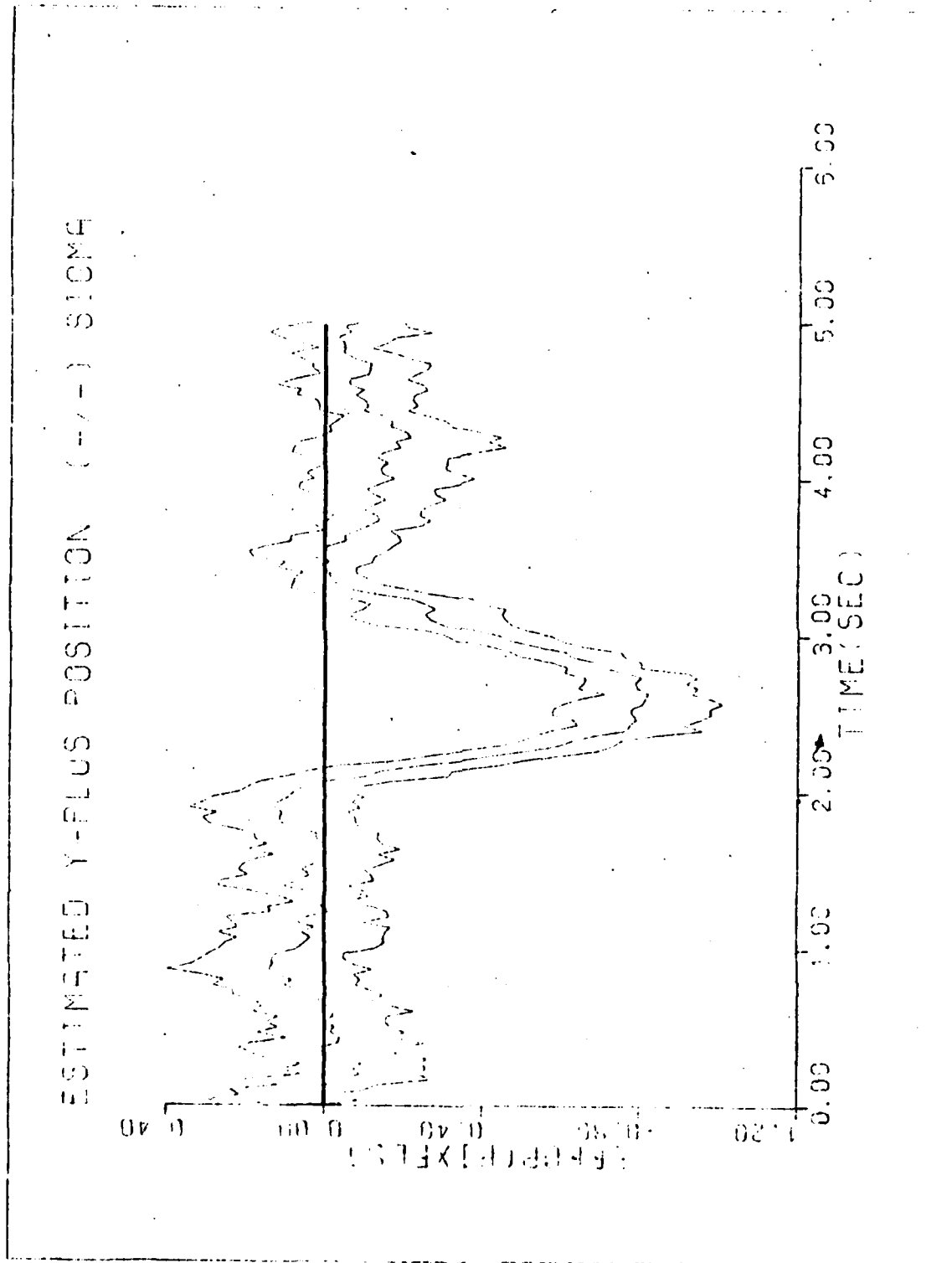


Figure C-12d. Case 12

ESTIMATED X-RAY CENTROID POSITION (+/-) SIGMA

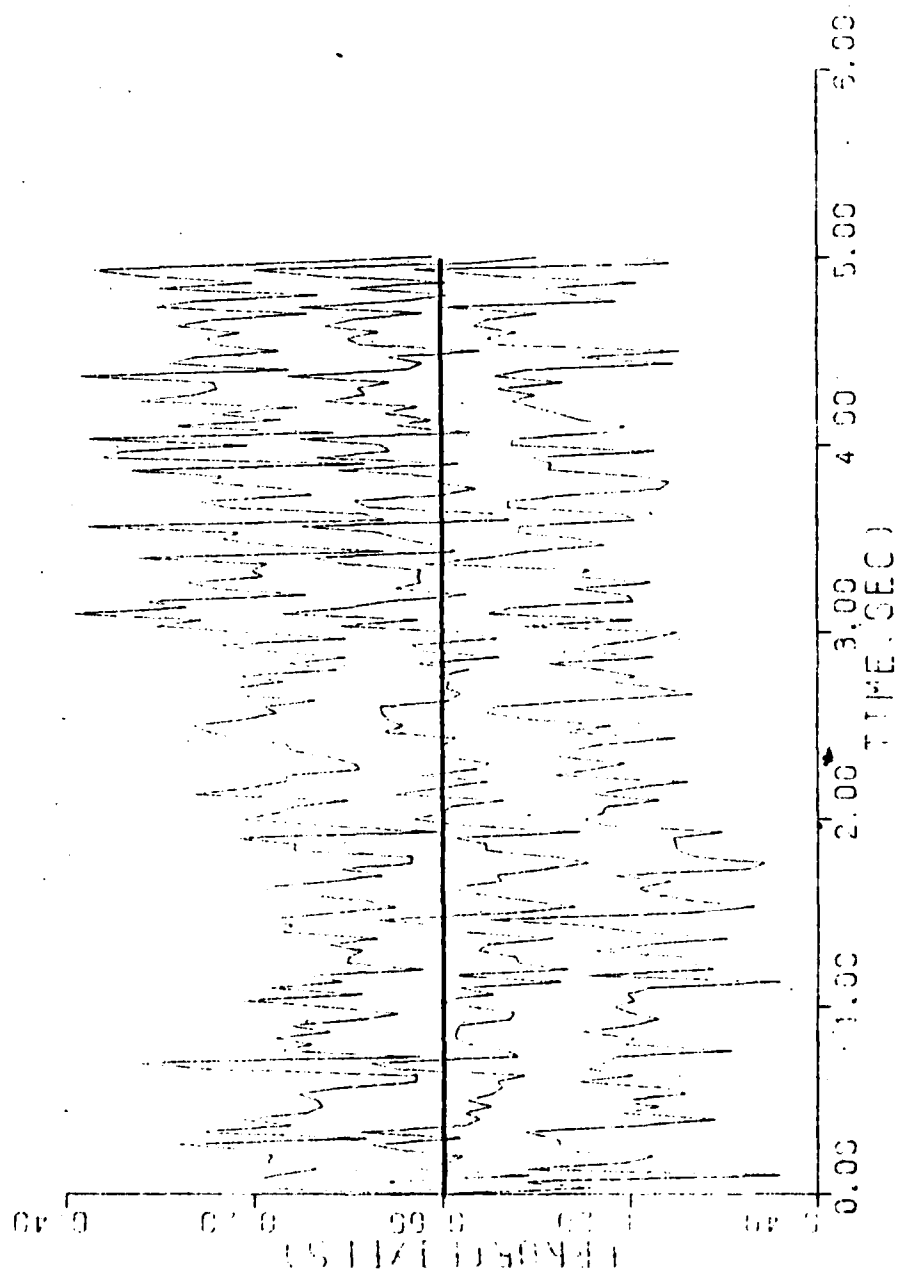


Figure C-12e. Case 12

9

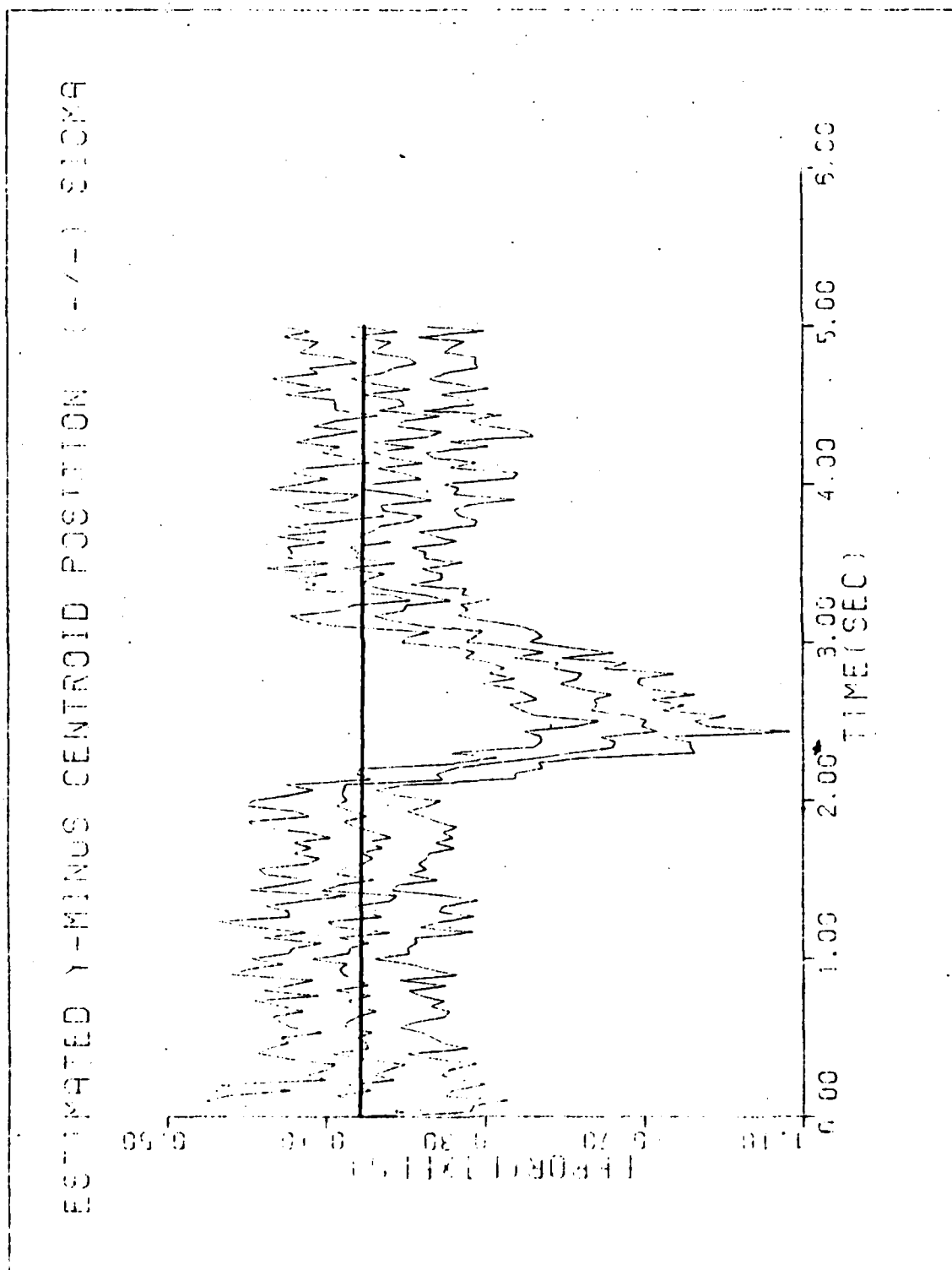


Figure C-12f. Case 12

ESTIMATED X-PLUS CENTROID POSITION 1- -1 210MP

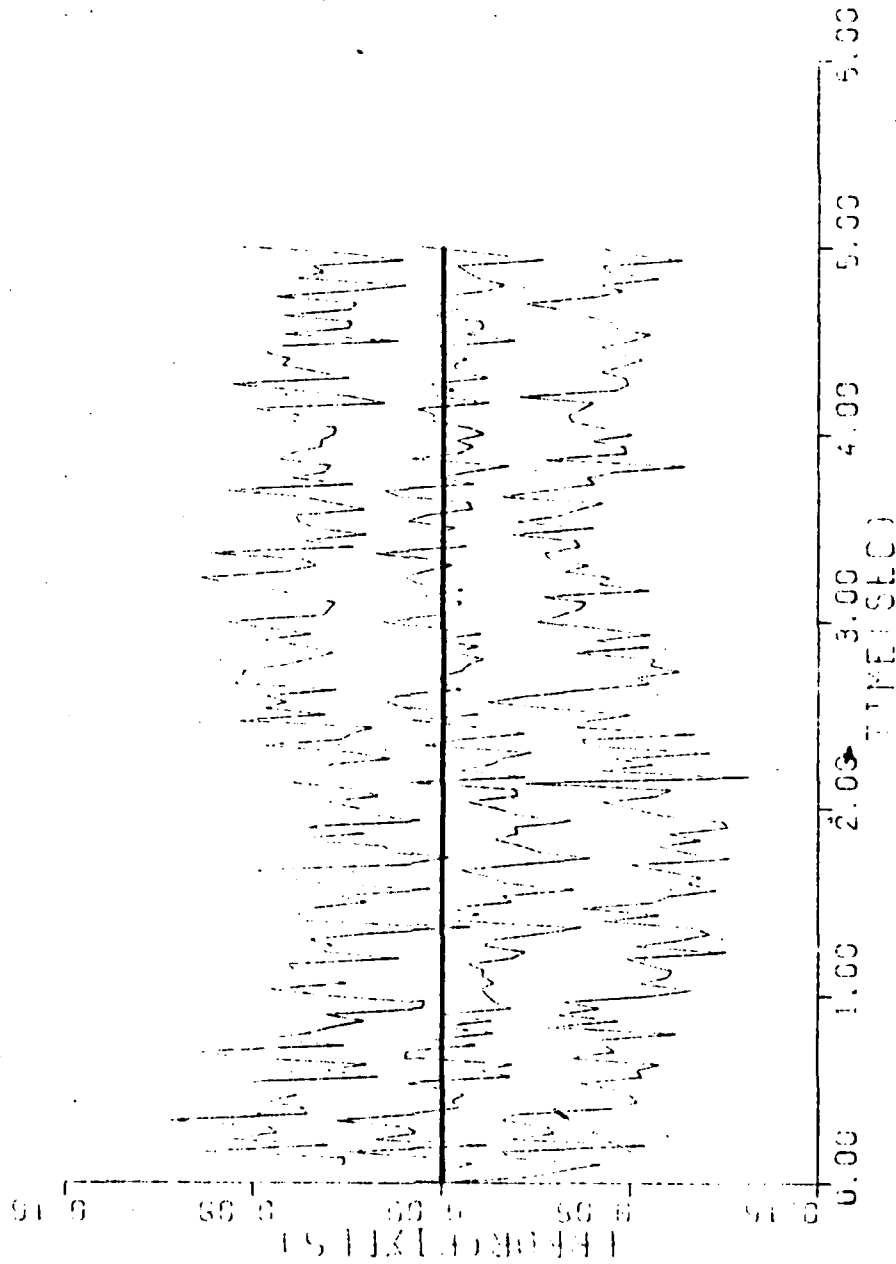


Figure C-12g. Case 12

9

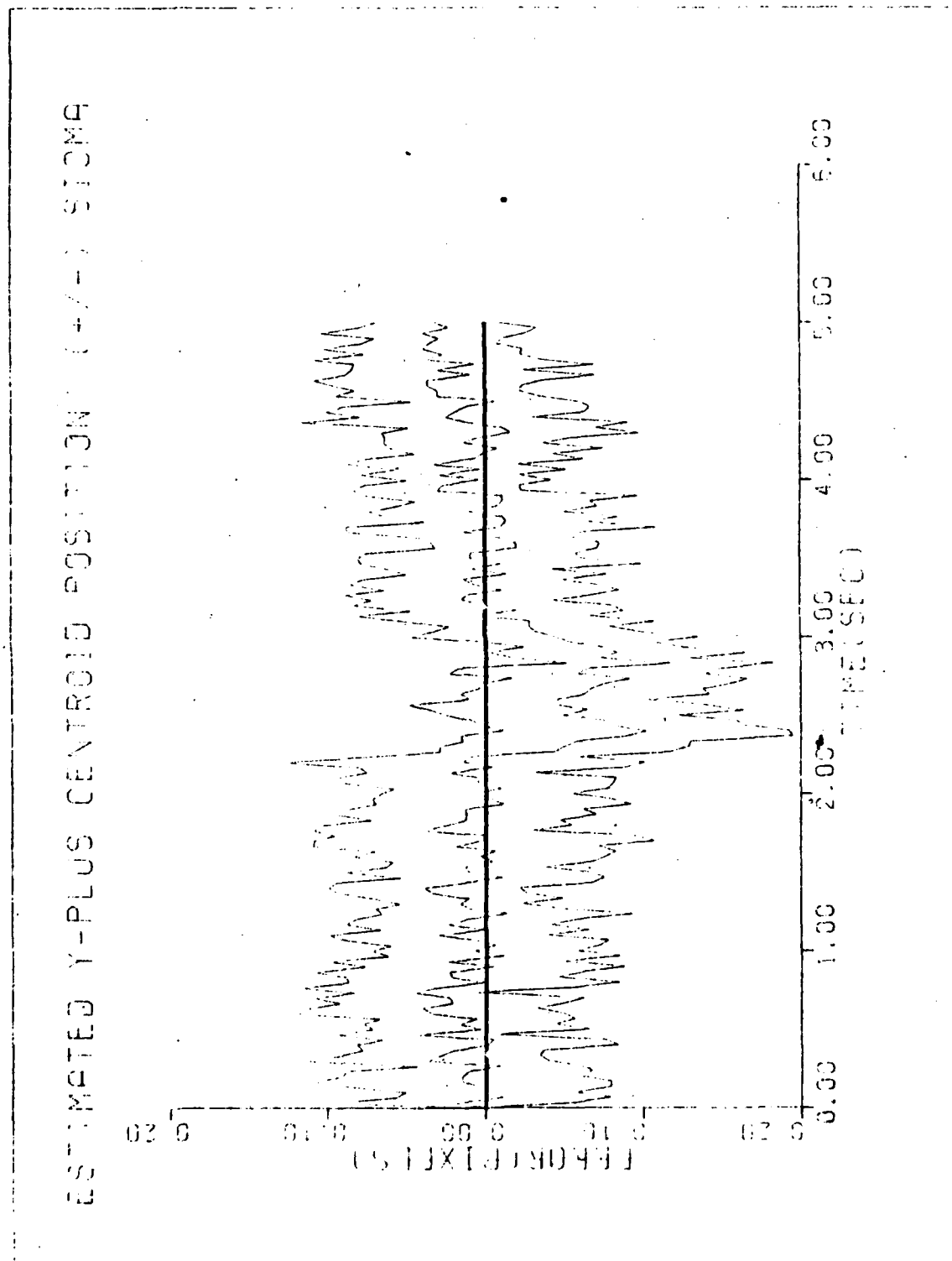


Figure C-12h. Case 12

The performance plots for this case were similar to the plots shown for case 12, and thus are omitted.

Figure C-13. Case 13

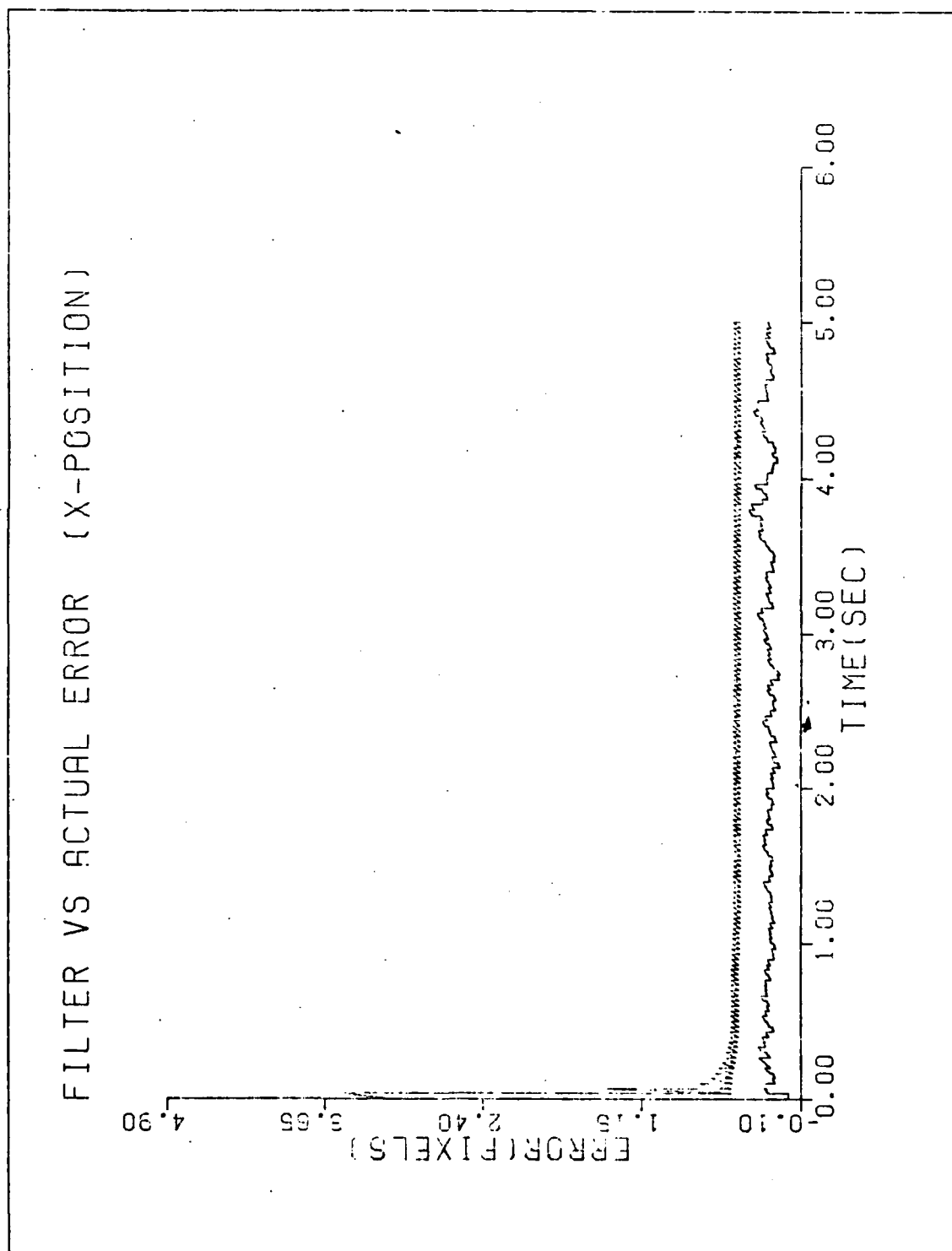


Figure C-14a. Case 14

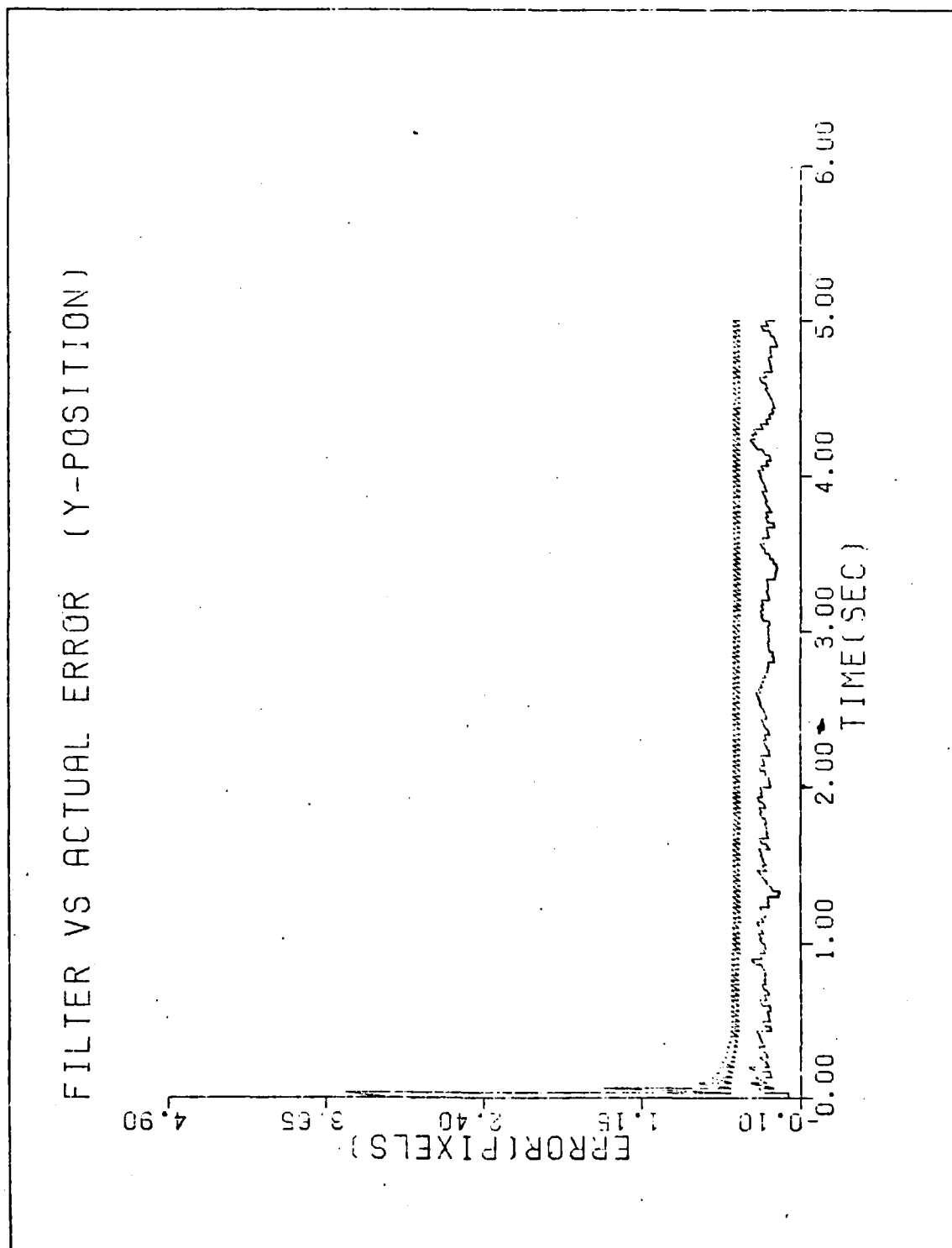


Figure C-14b. Case 14

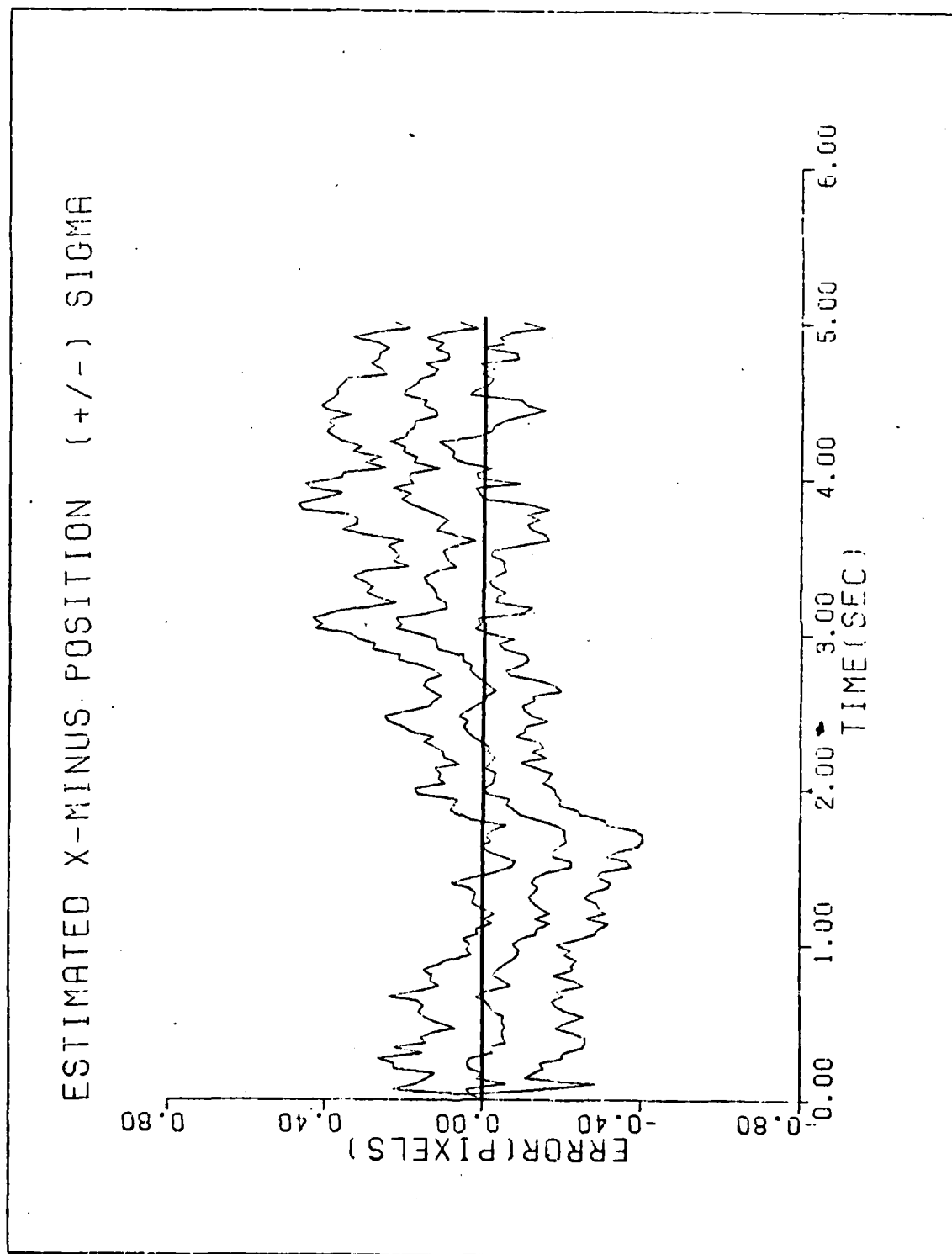


Figure C-14c. Case 14

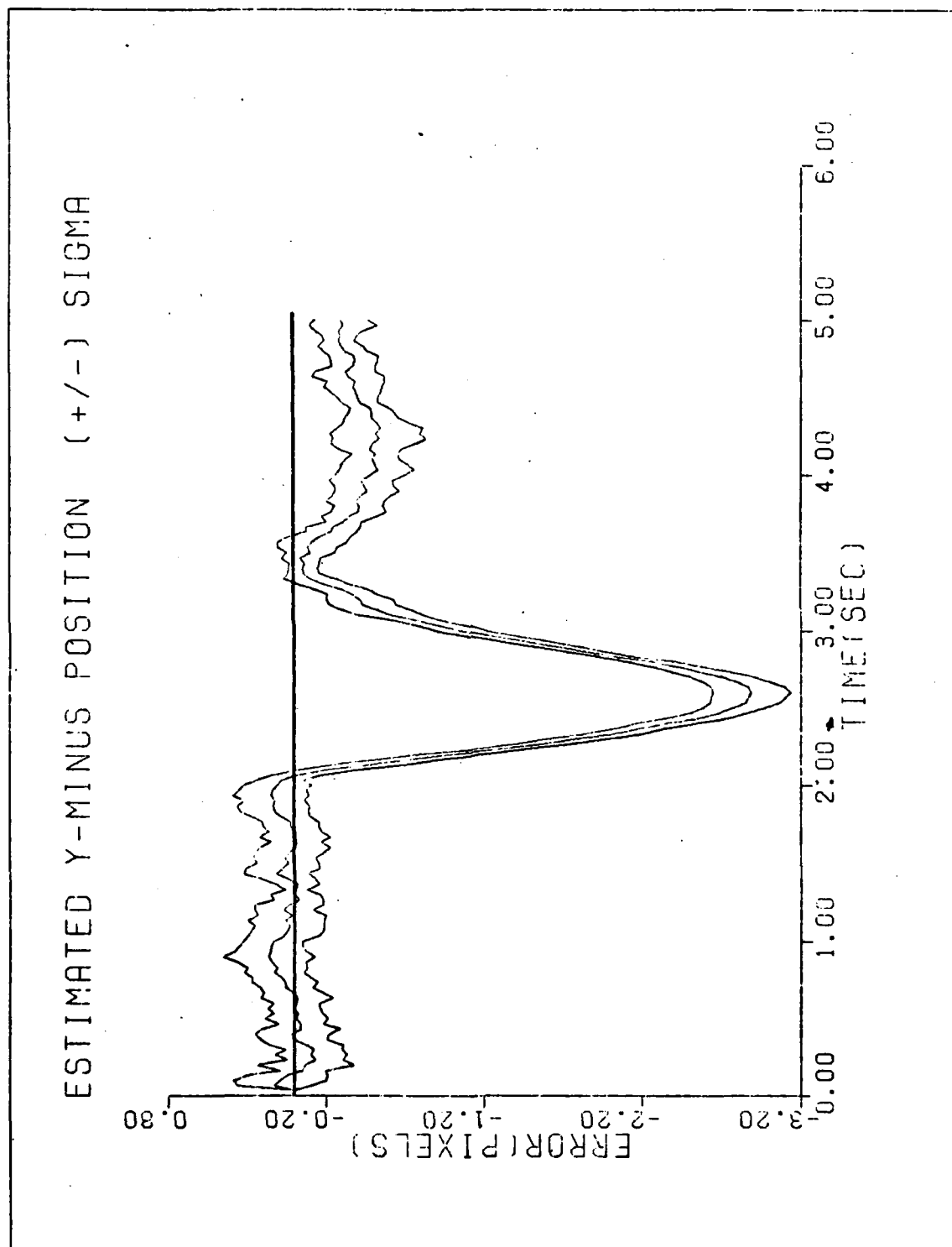


Figure C-14d. Case 14

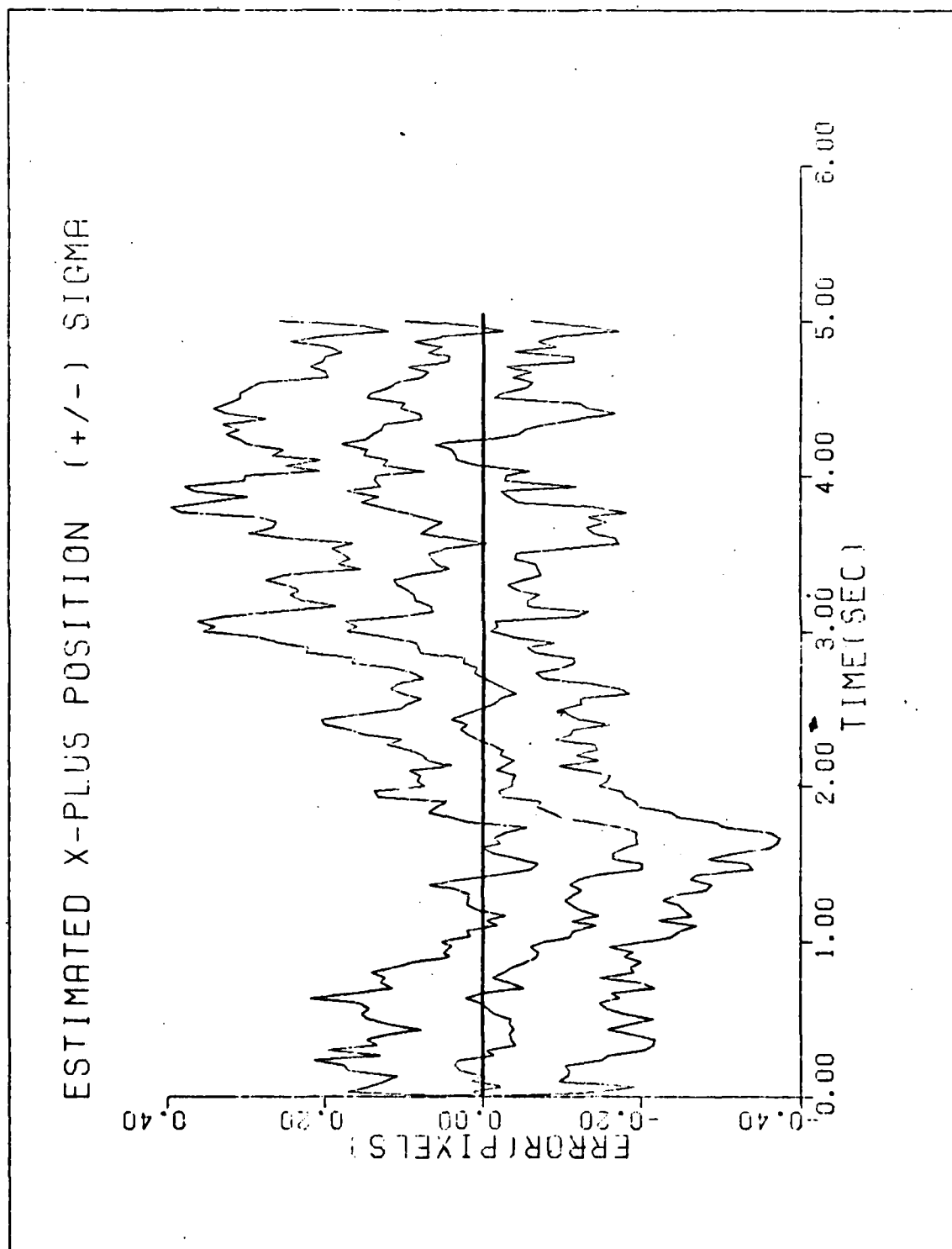


Figure C-14e. Case 14

ESTIMATED Y-PLUS POSITION (+/-) SIGMA

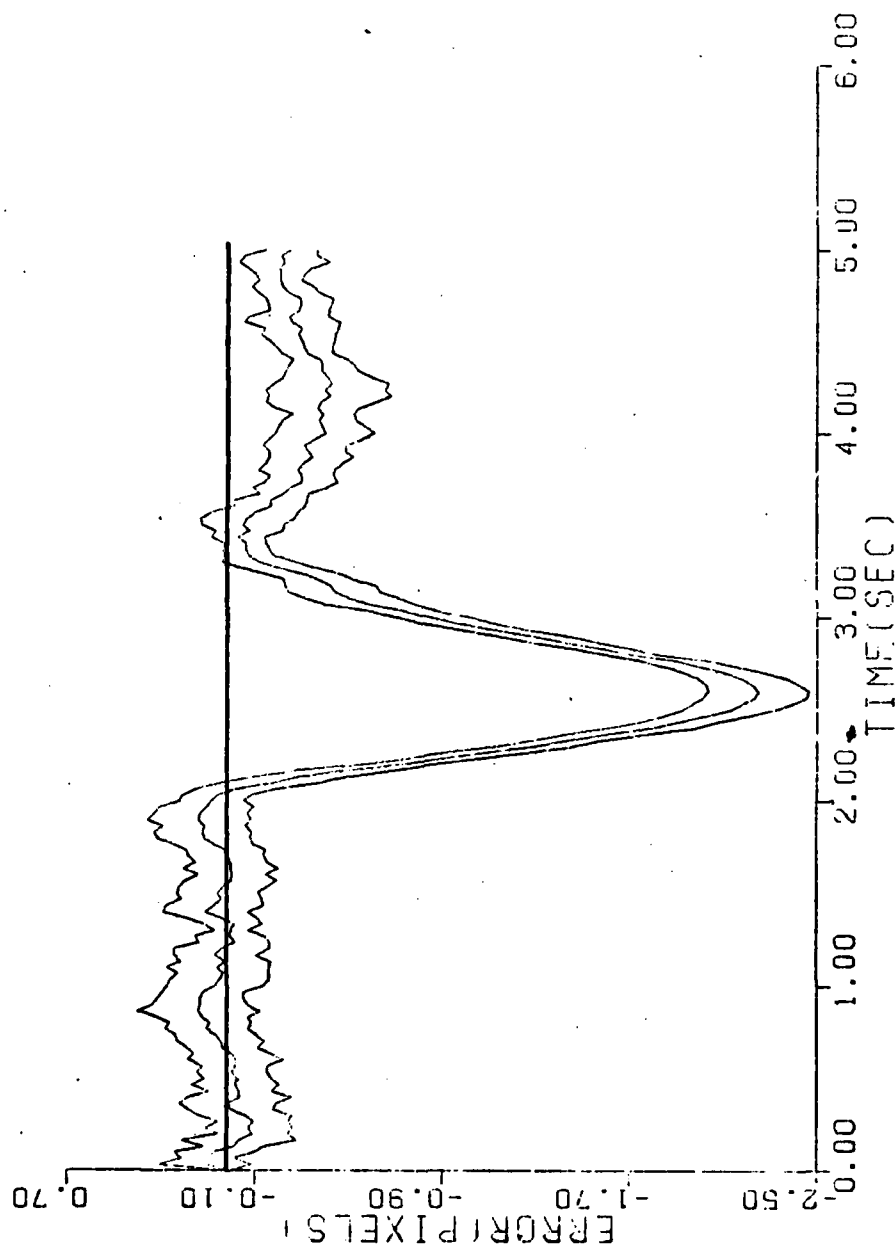


Figure C-14f. Case 14

ESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA

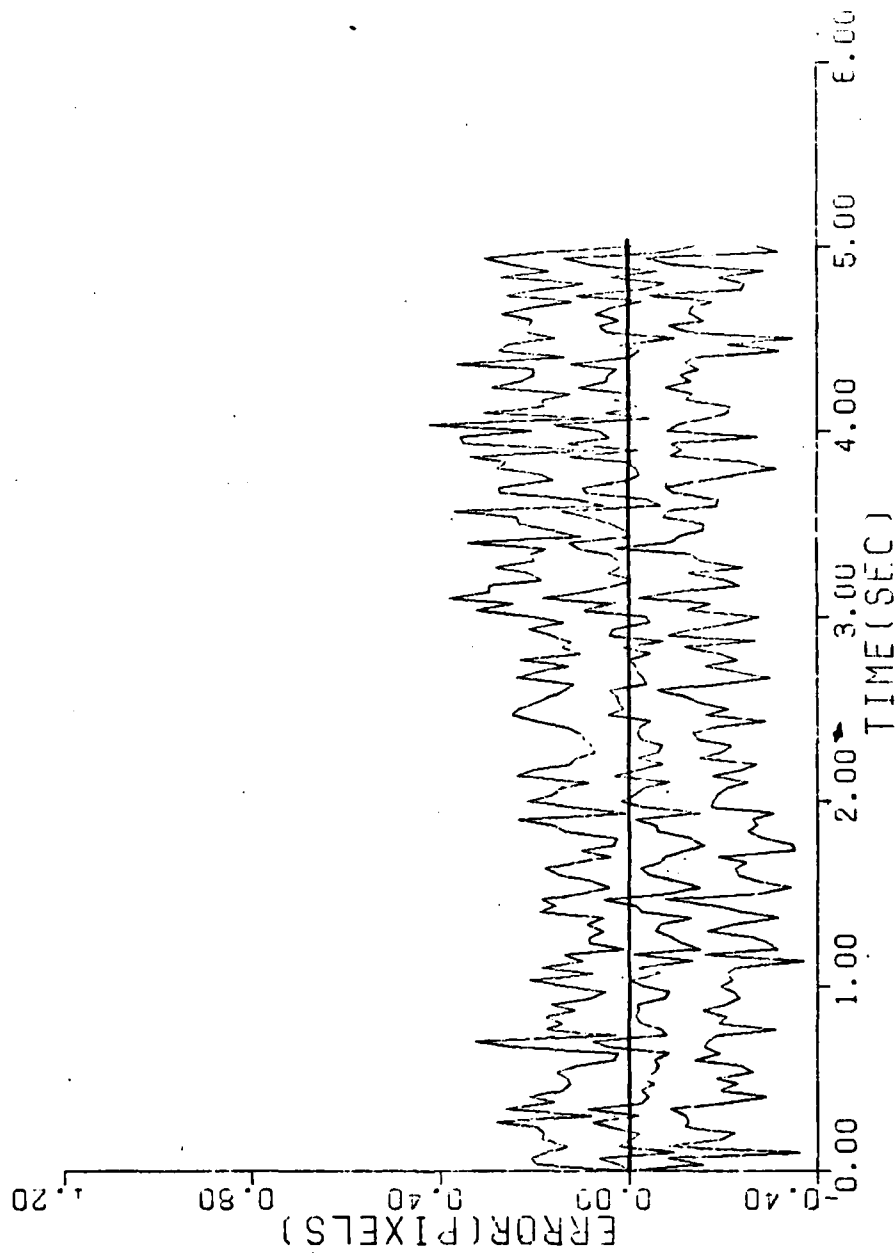


Figure C-14g. Case 14

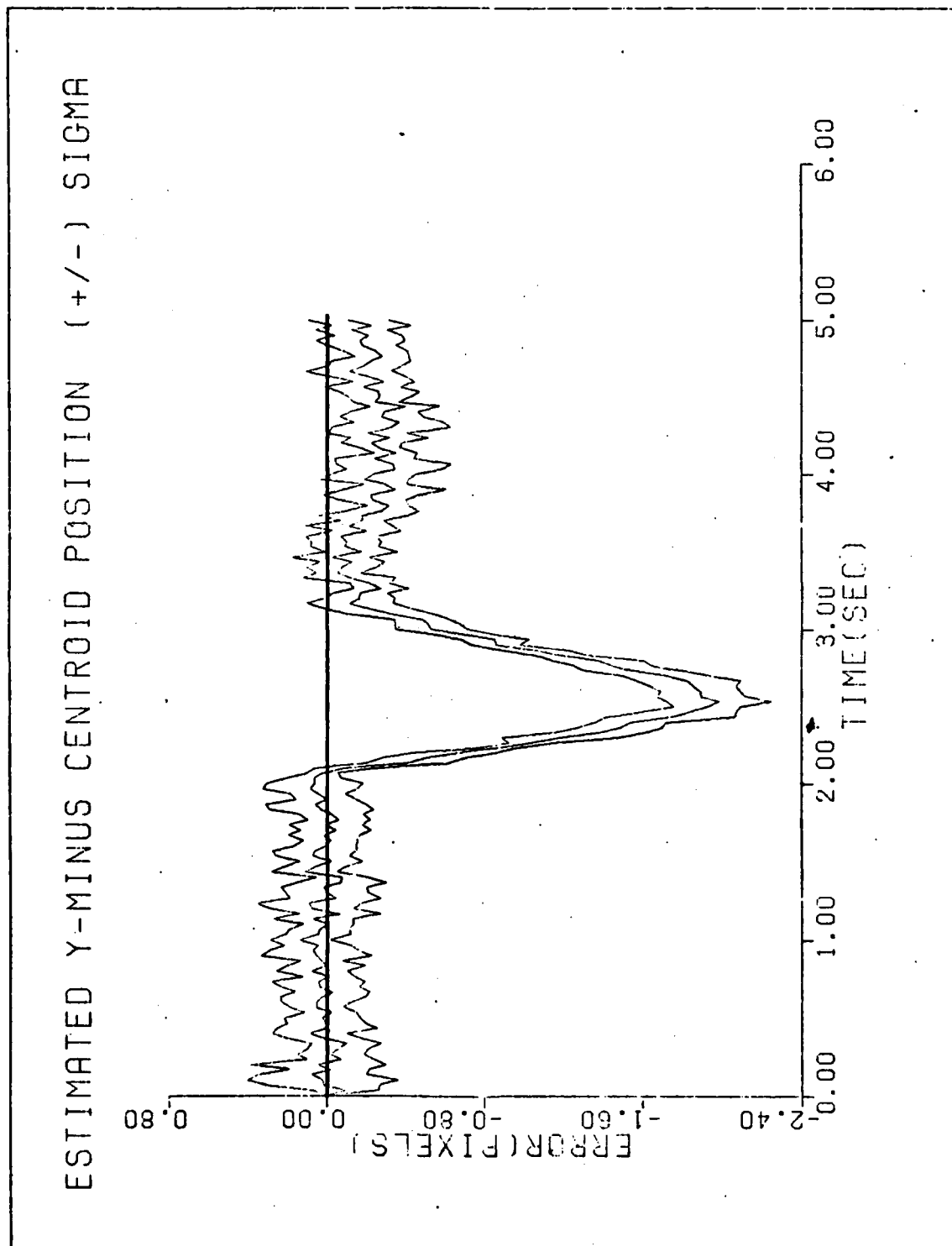


Figure C-14h. Case 14

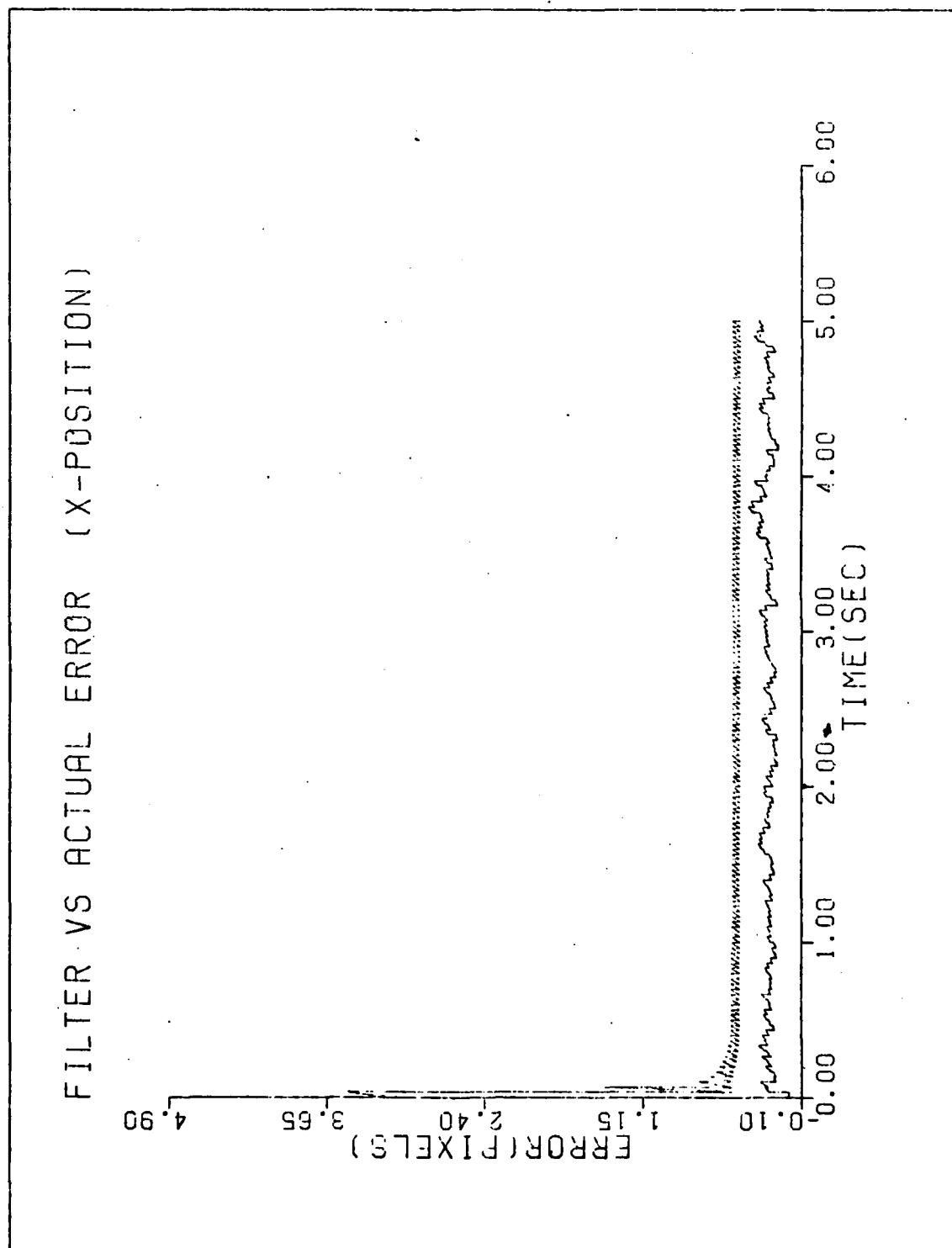


Figure C-15a. Case 15

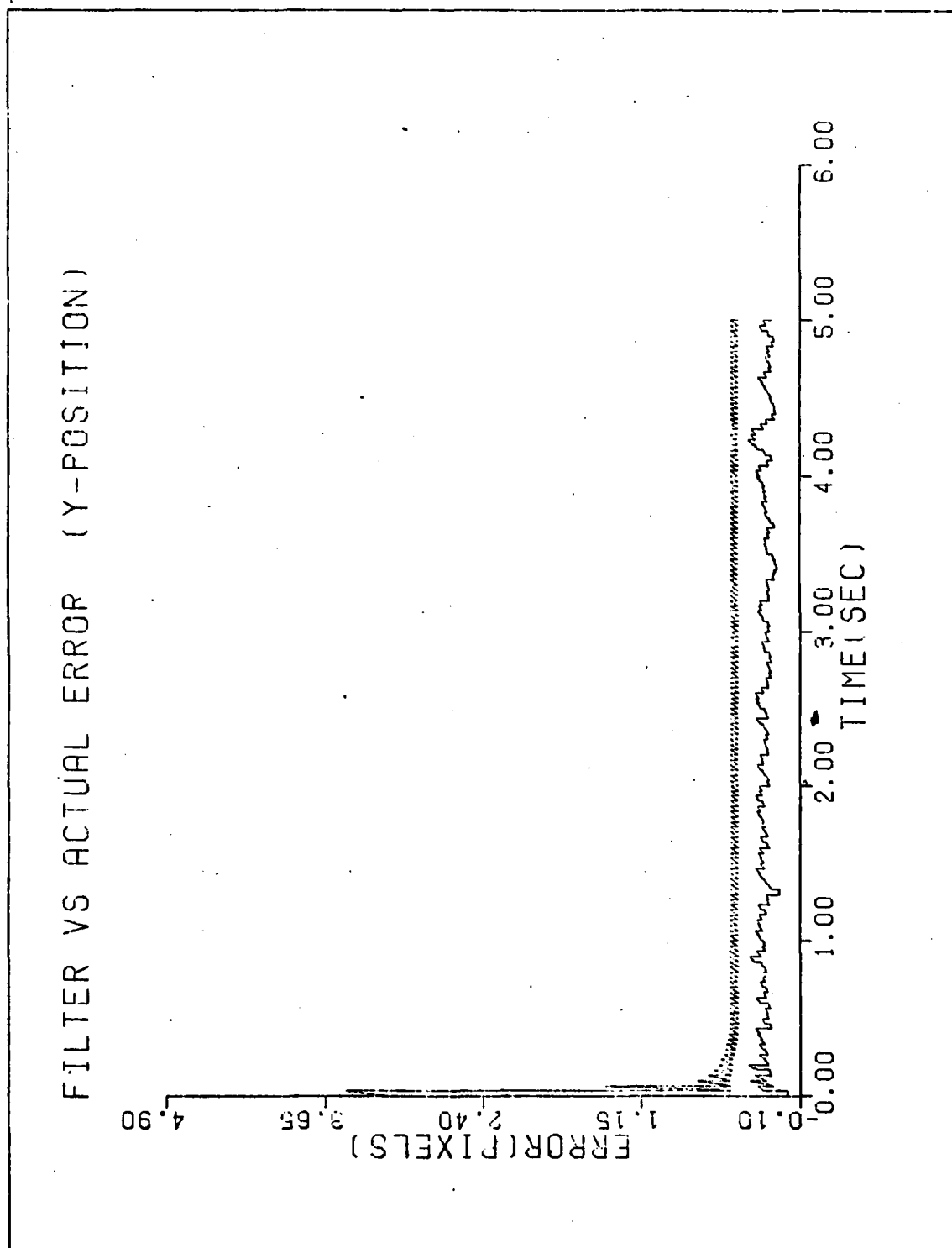


Figure C-15b. Case 15

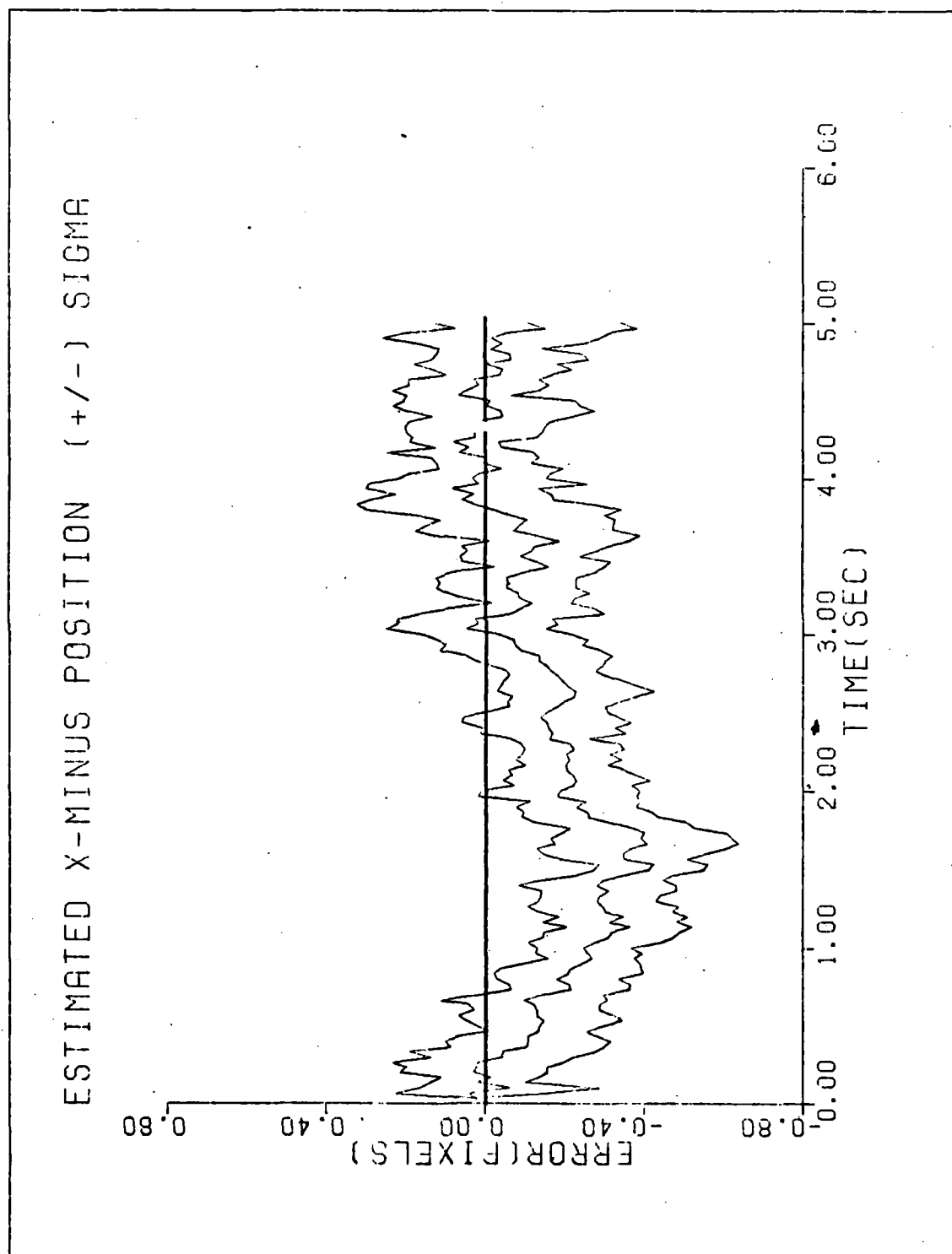


Figure C-15c. Case 15

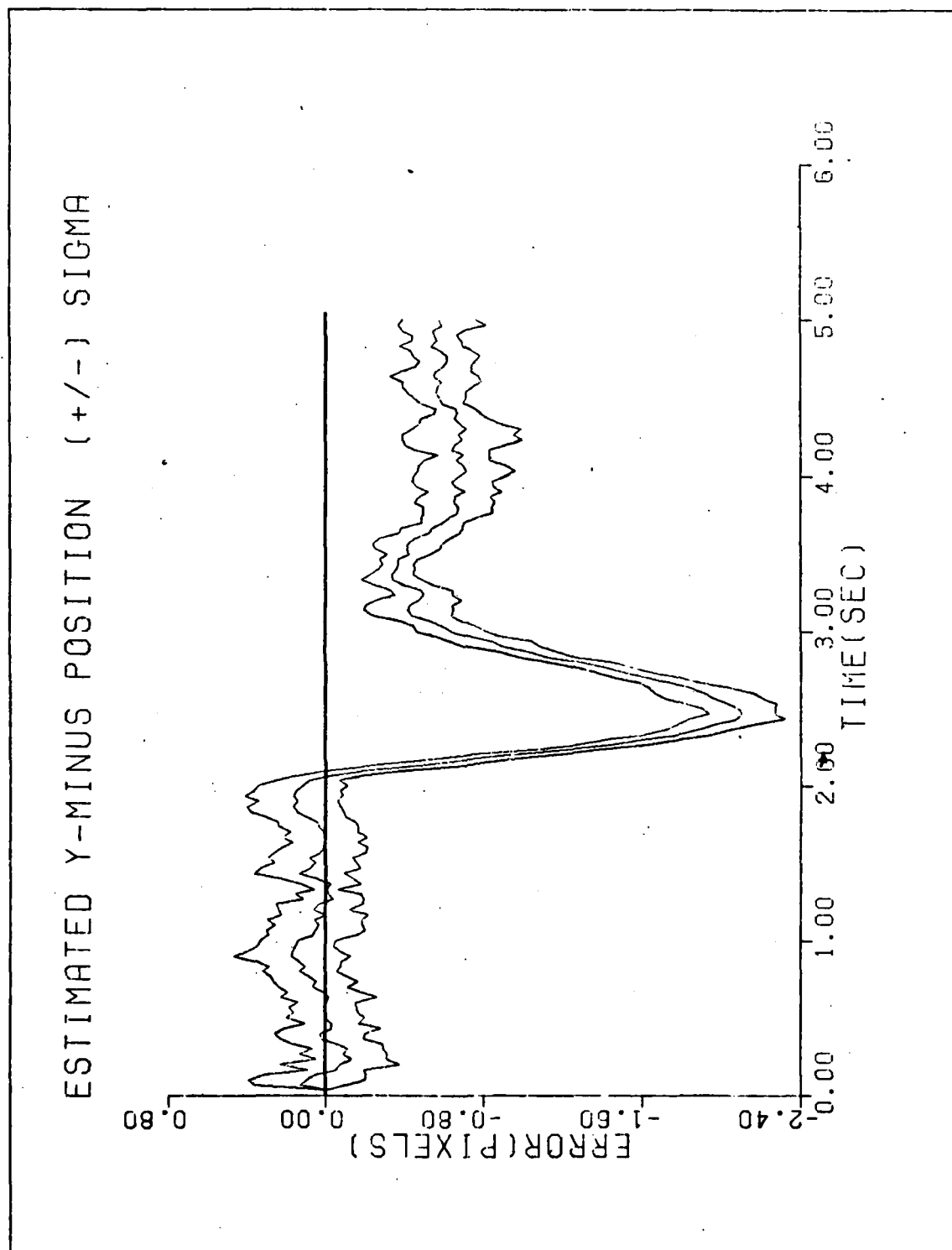


Figure C-15d. Case 15

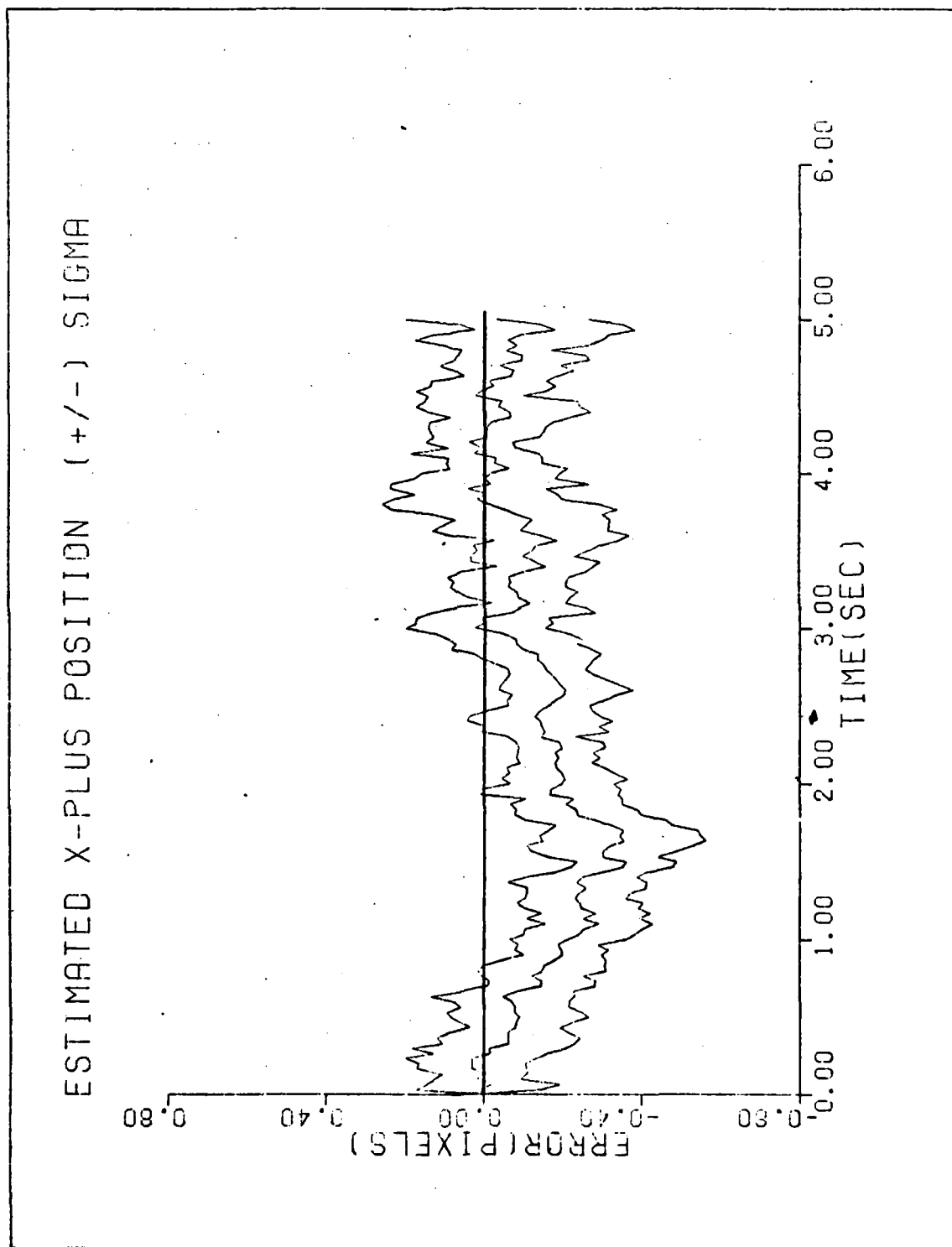


Figure C-15e. Case 15

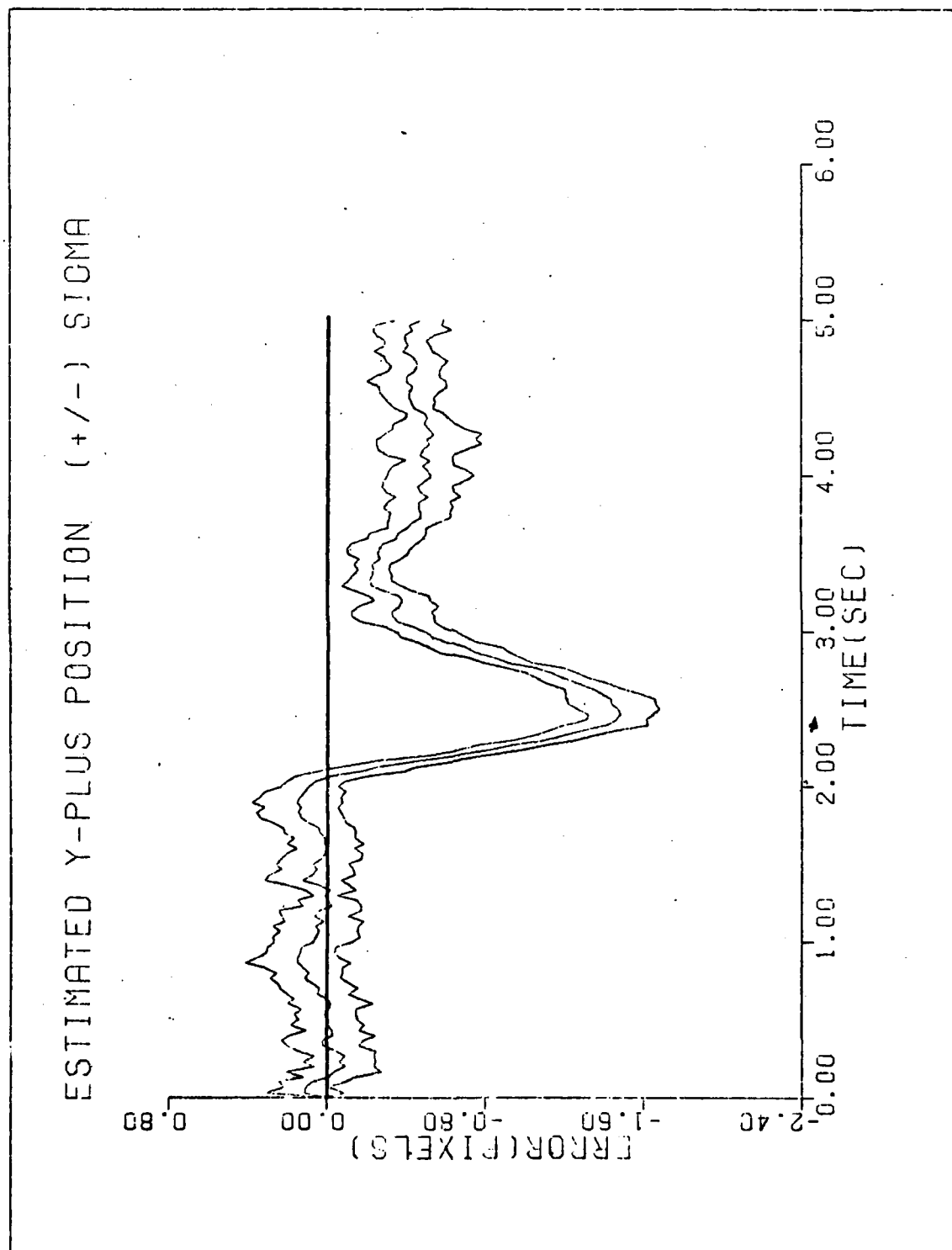


Figure C-15f. Case 15

ESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA

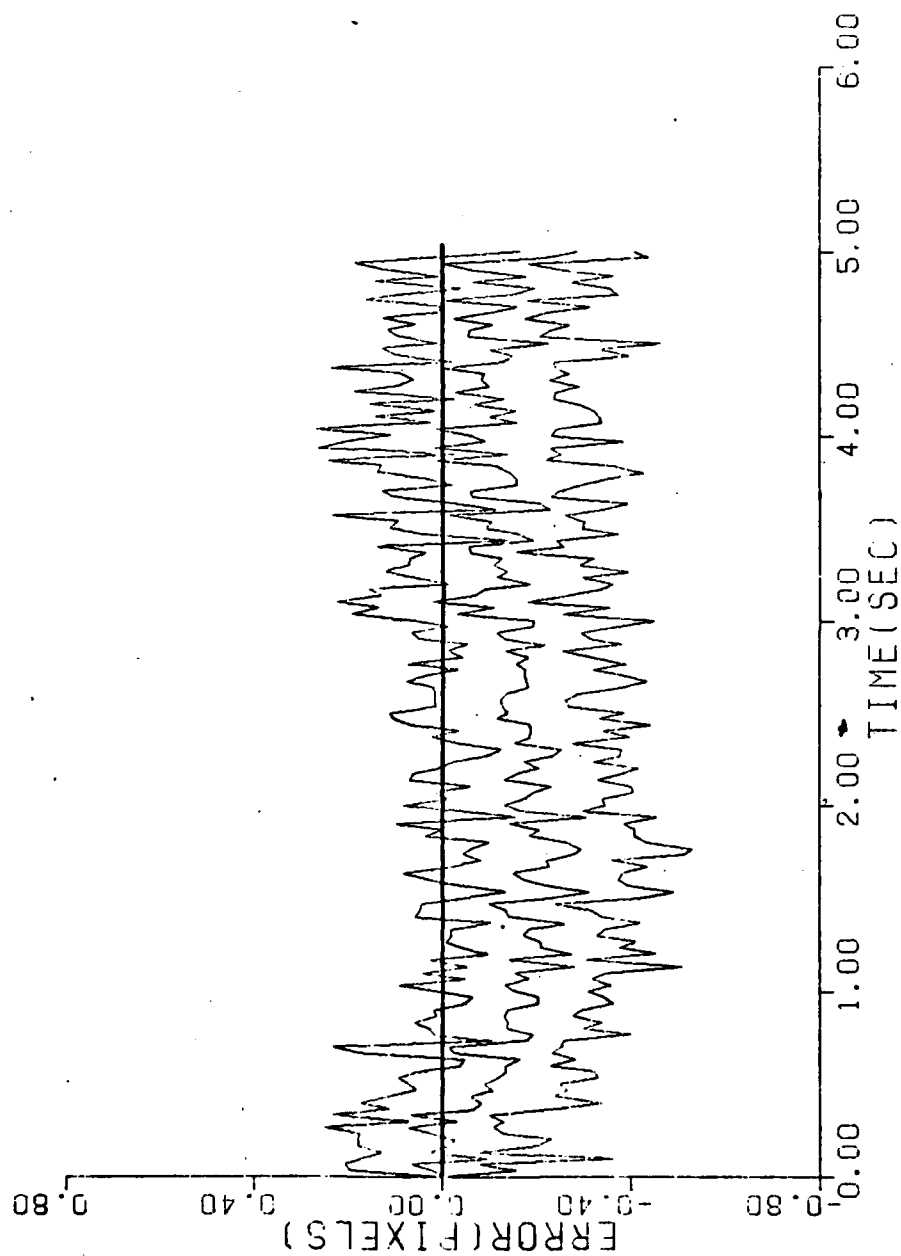


Figure C-15g. Case 15

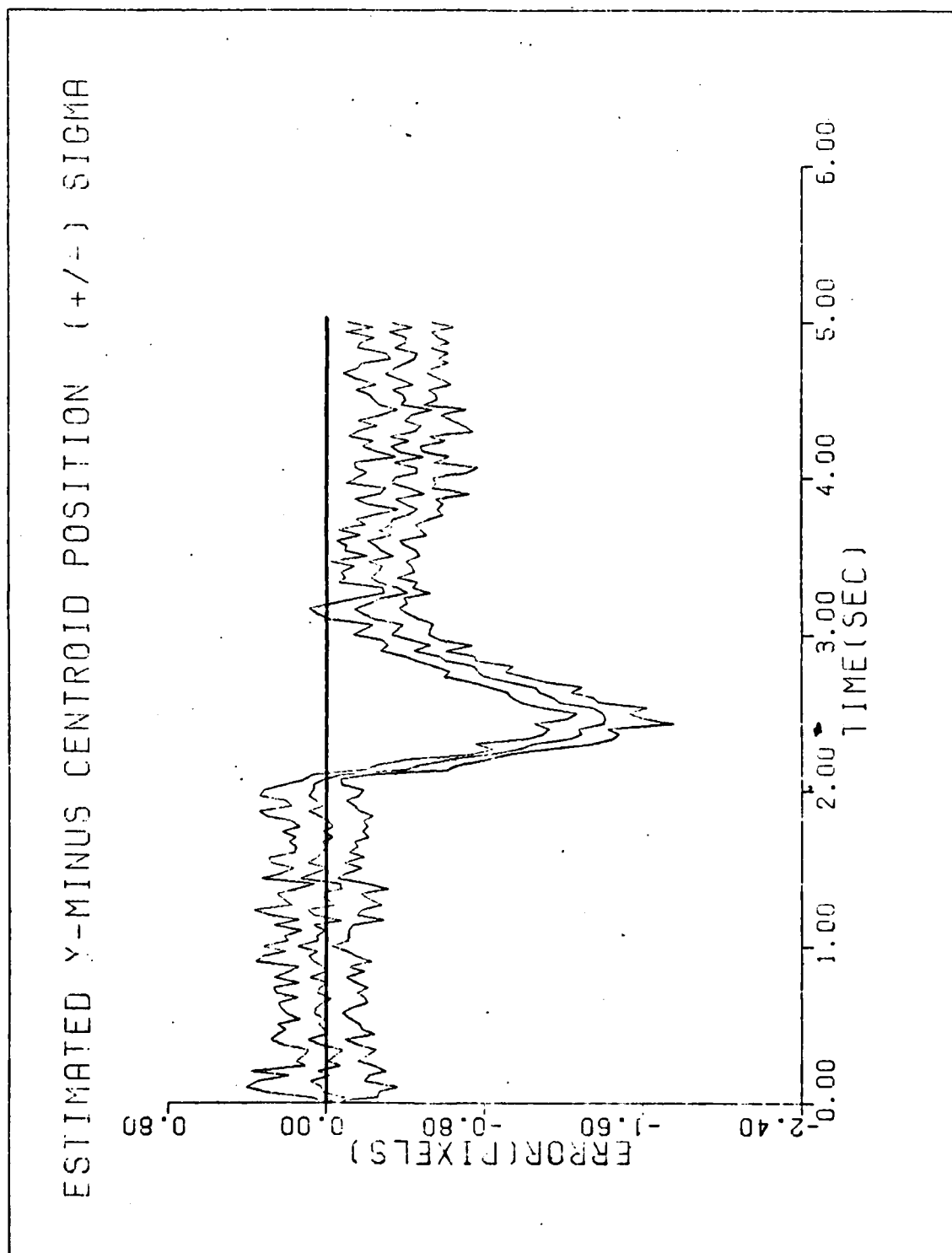


Figure C-15h. Case 15

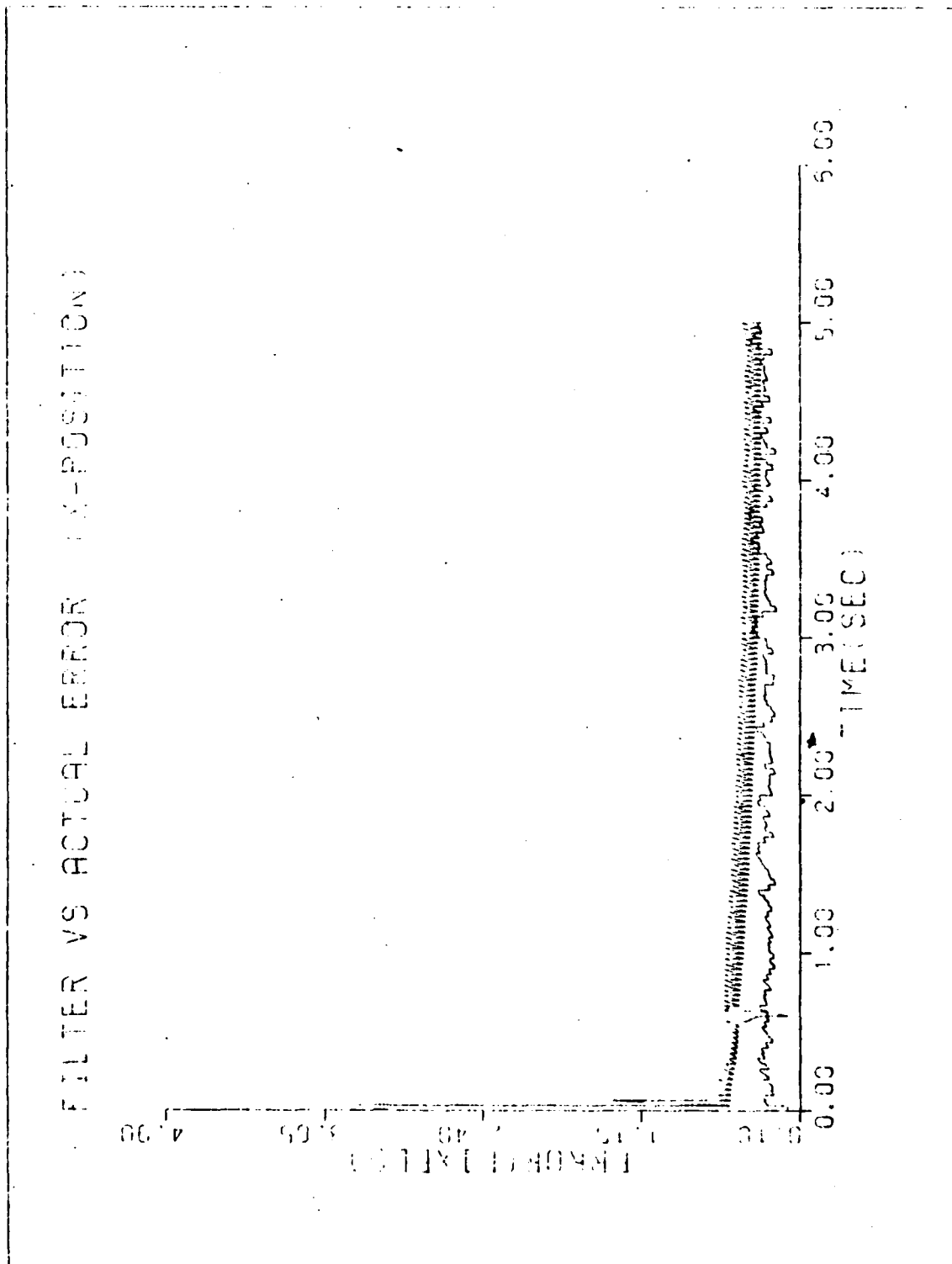


Figure C-16a. Case 16

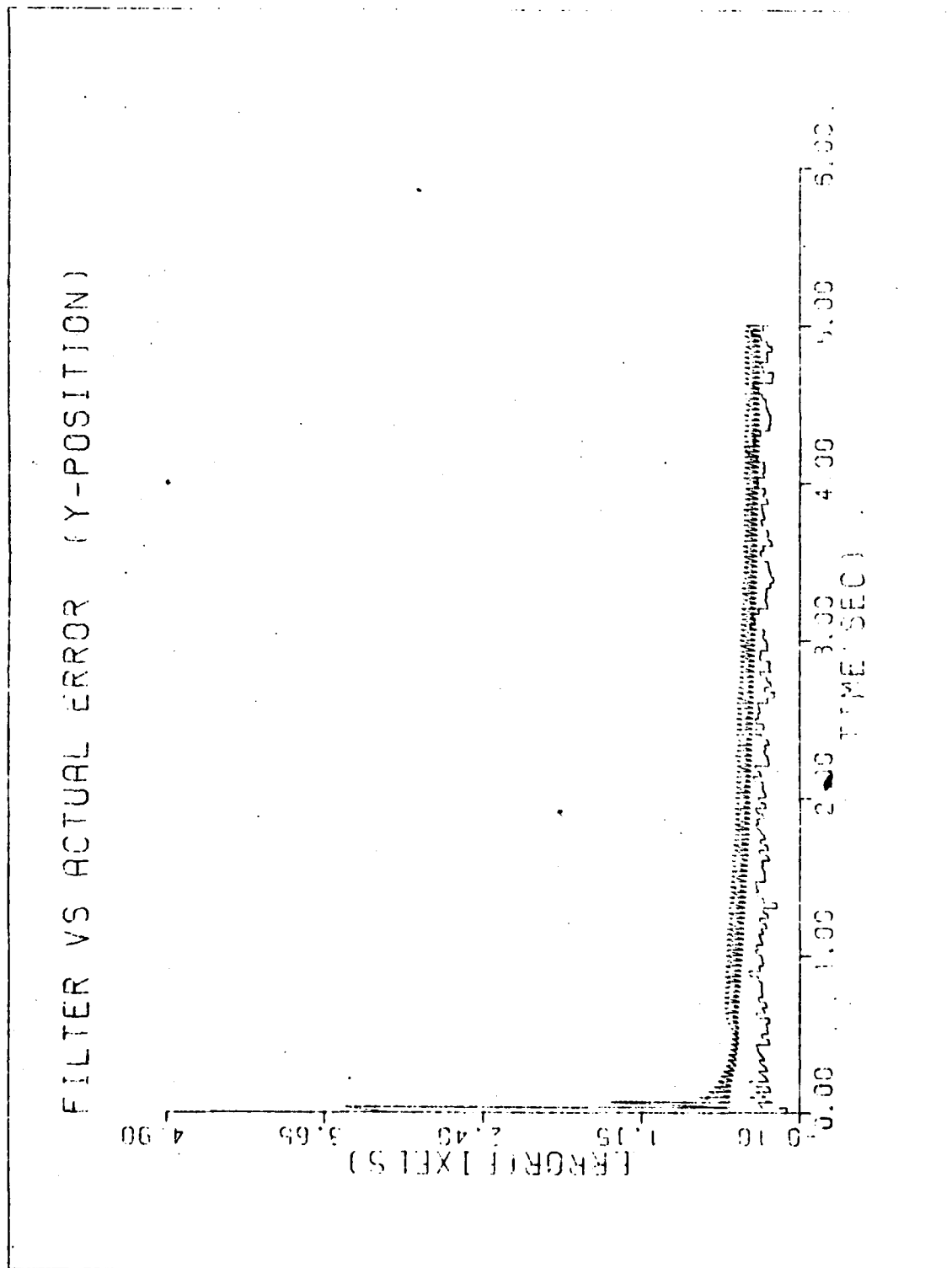


Figure C-16b. Case 16

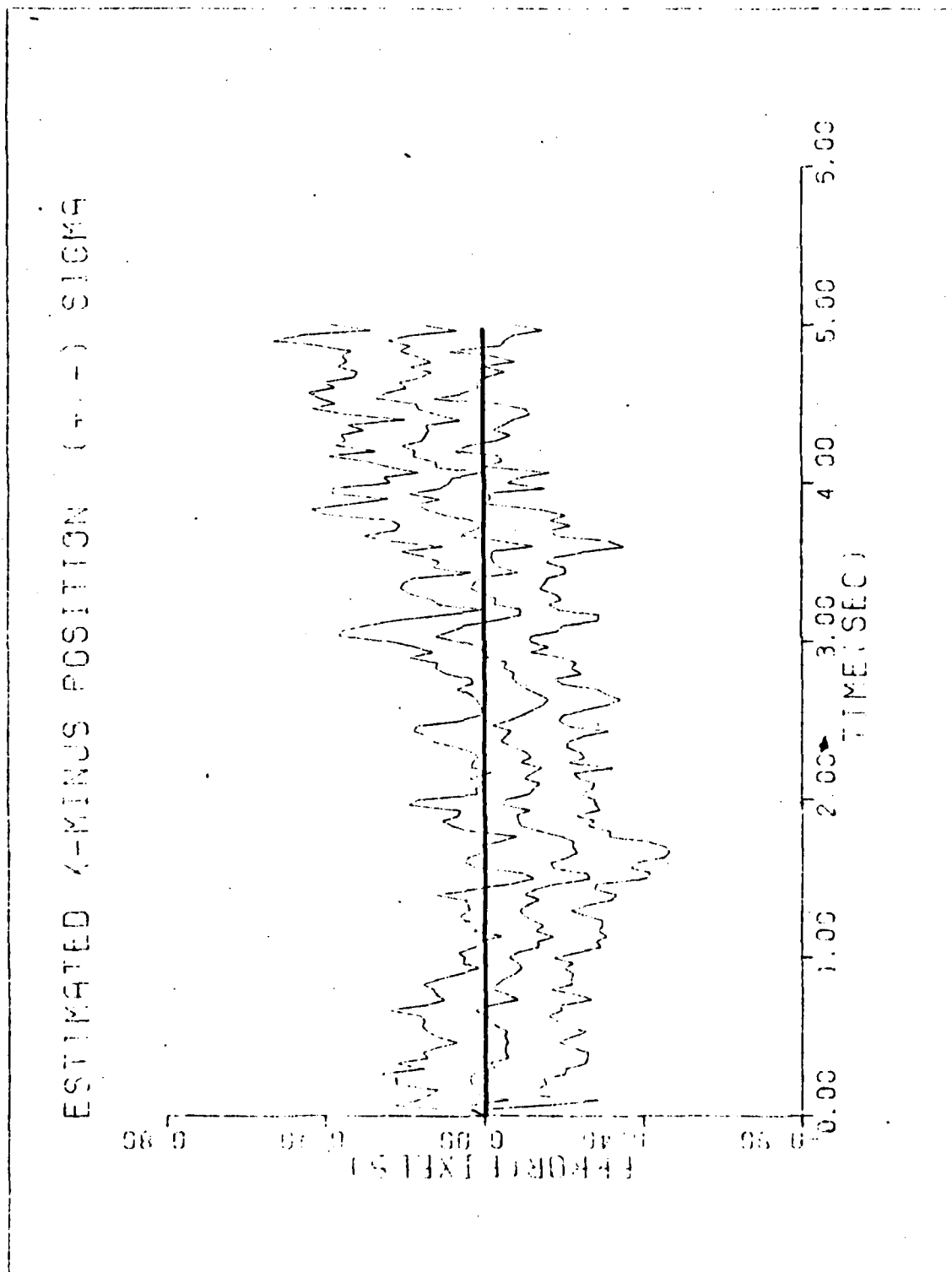


Figure C-16c. Case 16

9

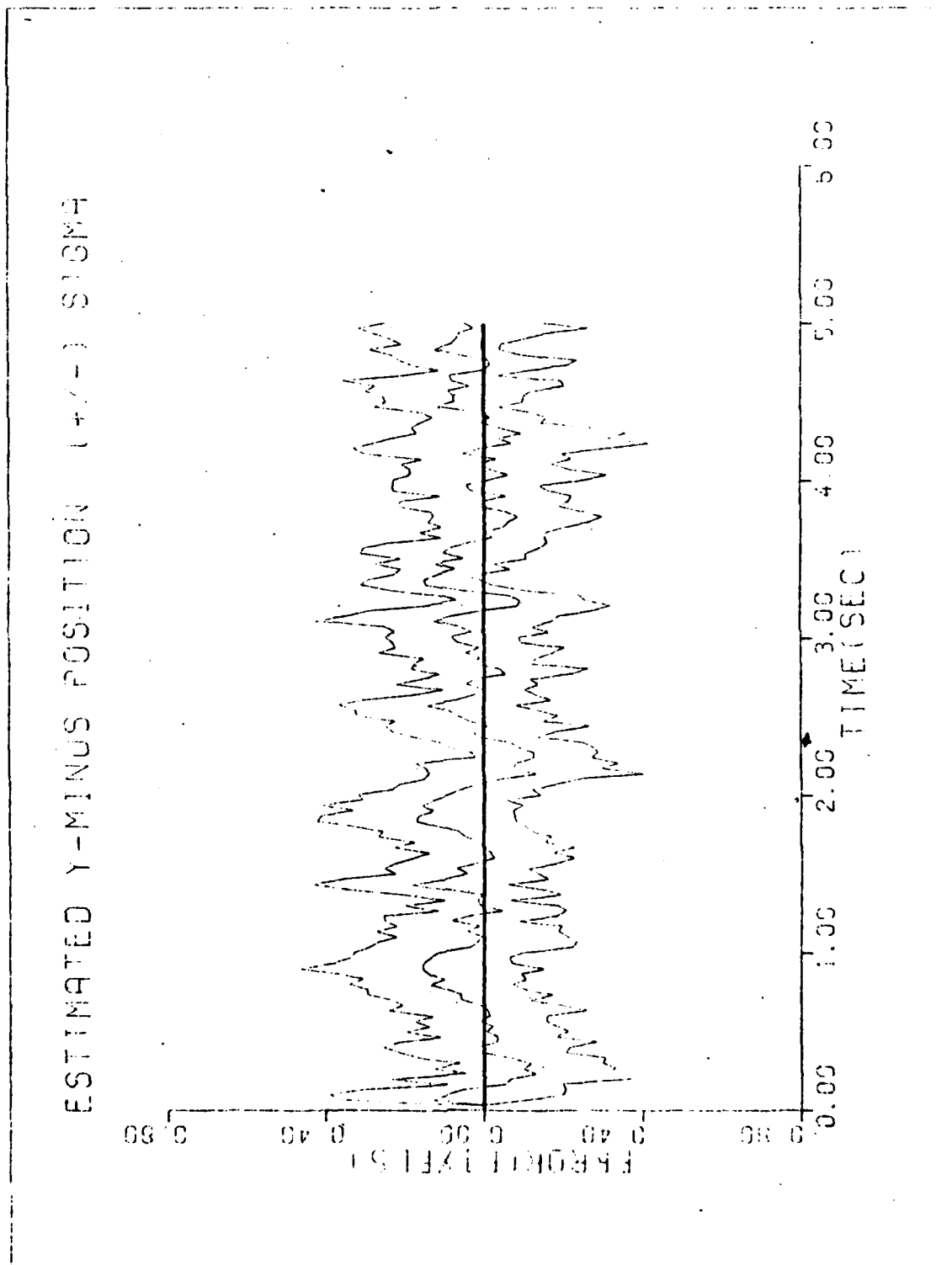


Figure C-16d. Case 16

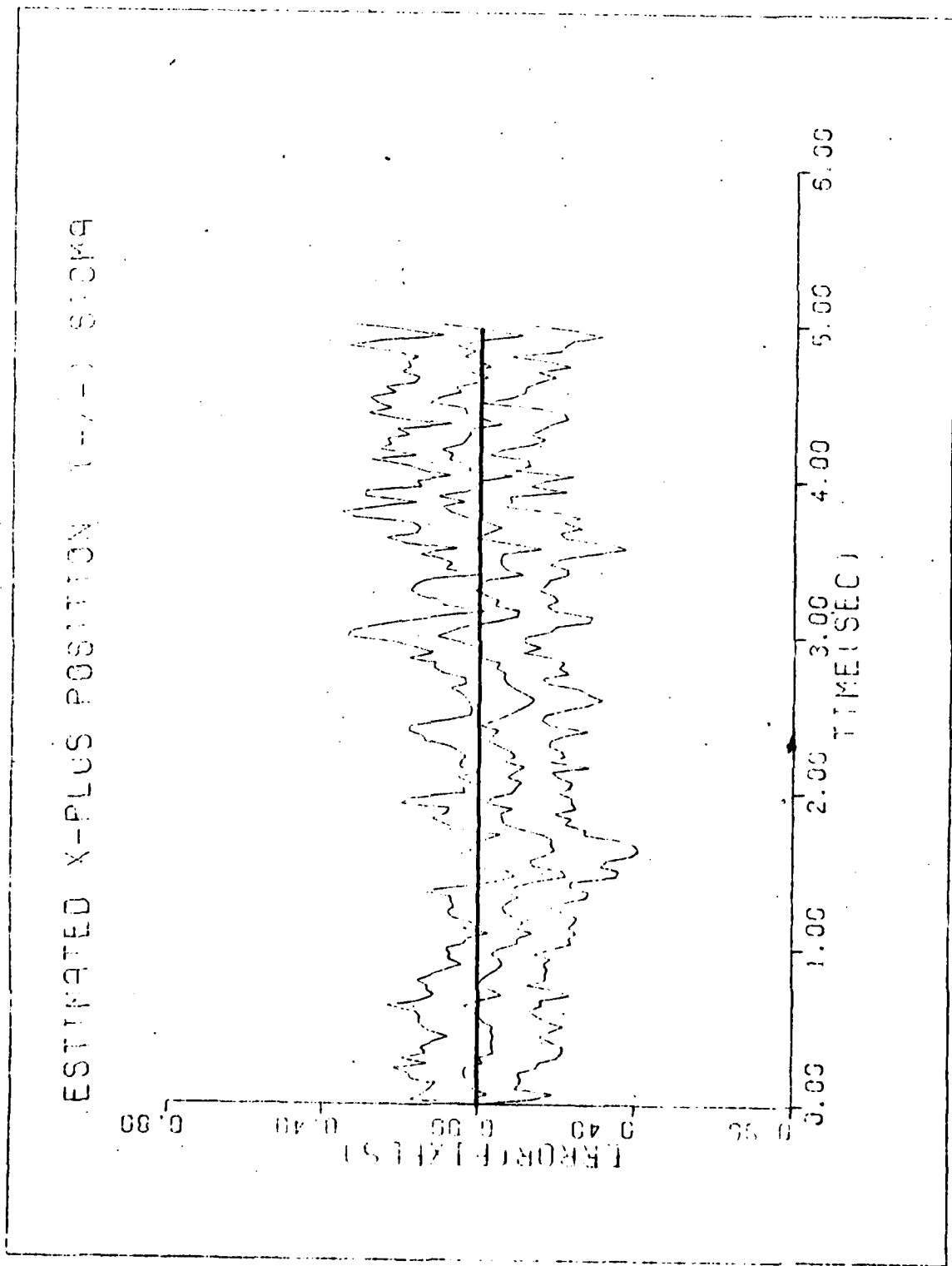


Figure C-16e. Case 16

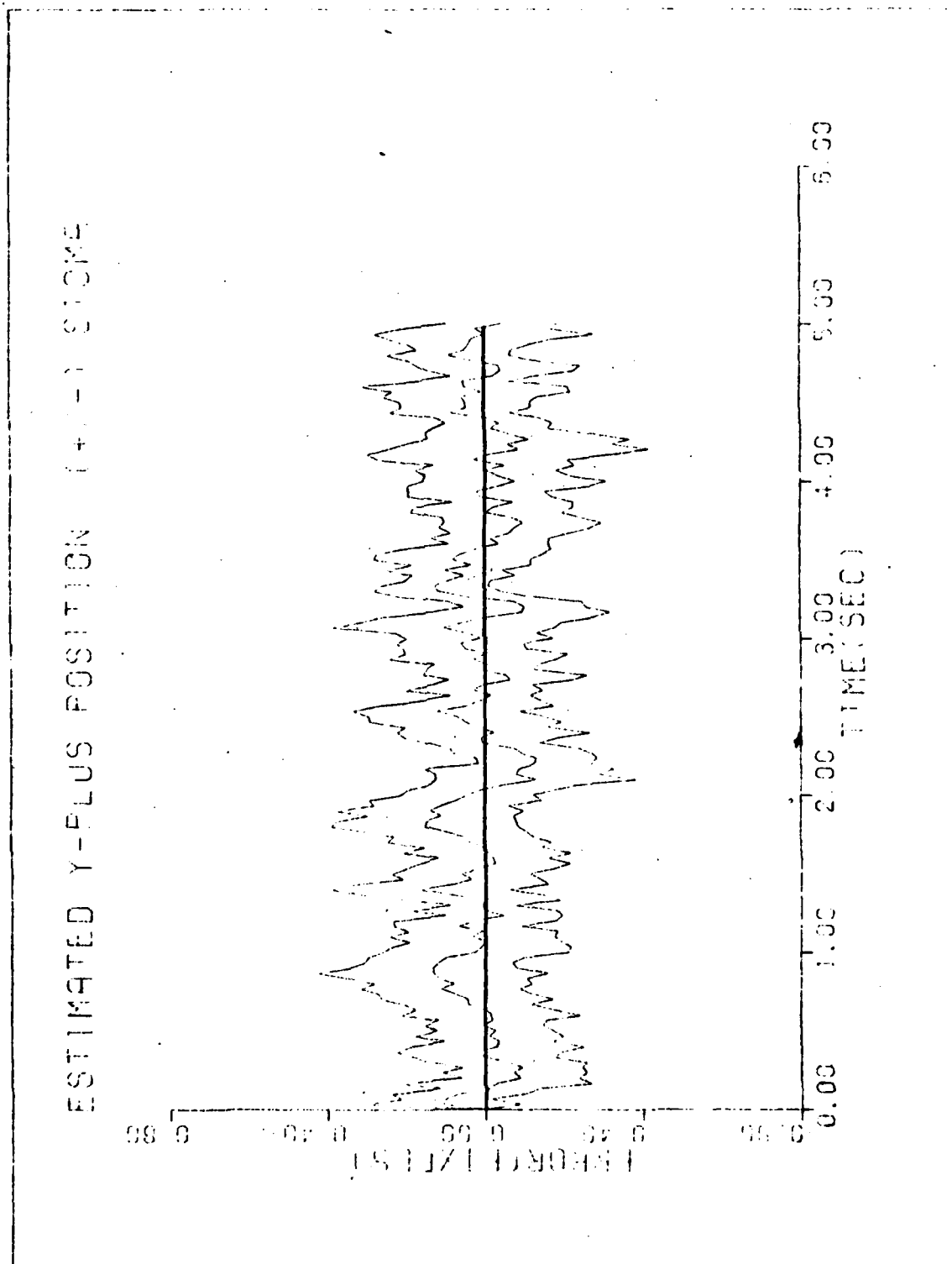


Figure C-16f. Case 16

ESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA

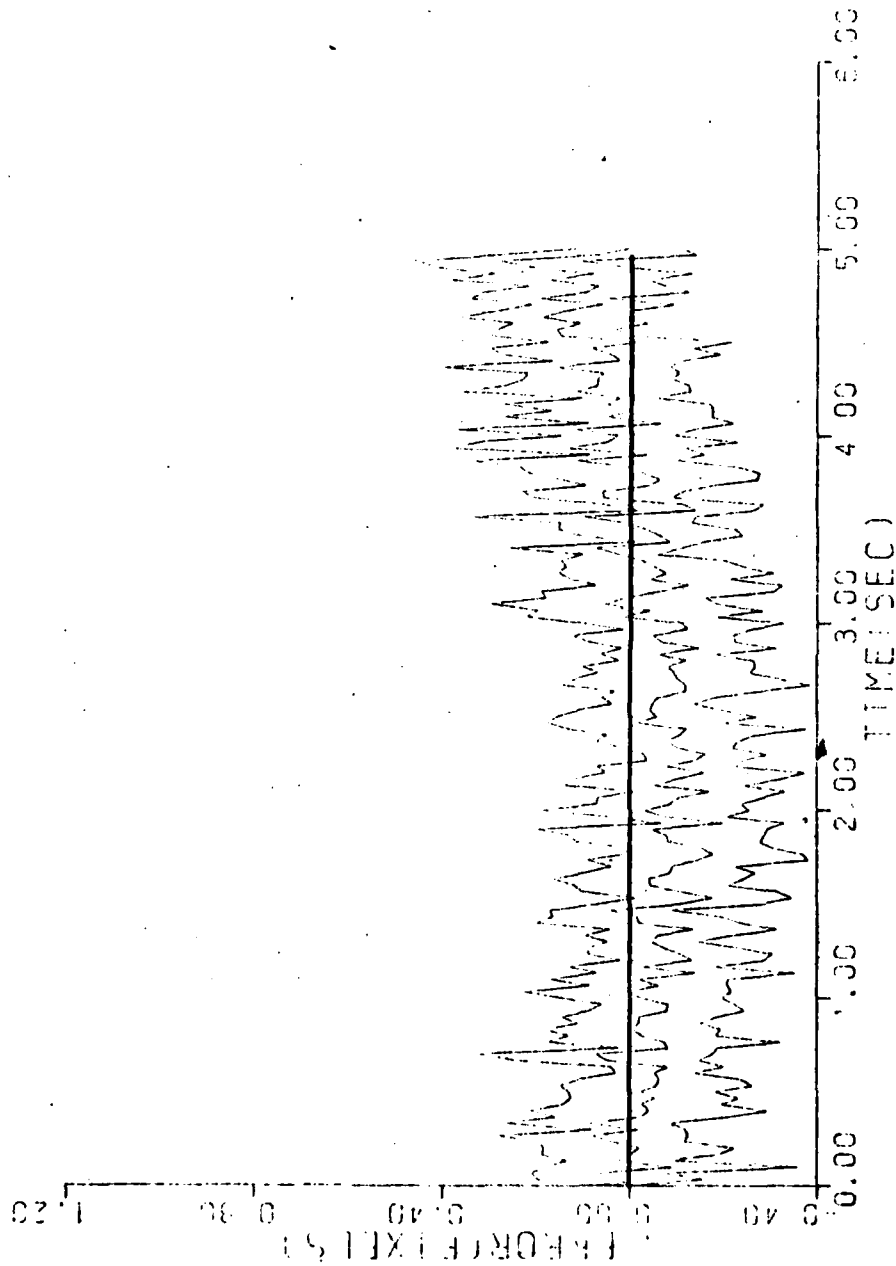


Figure C-16g. Case 16

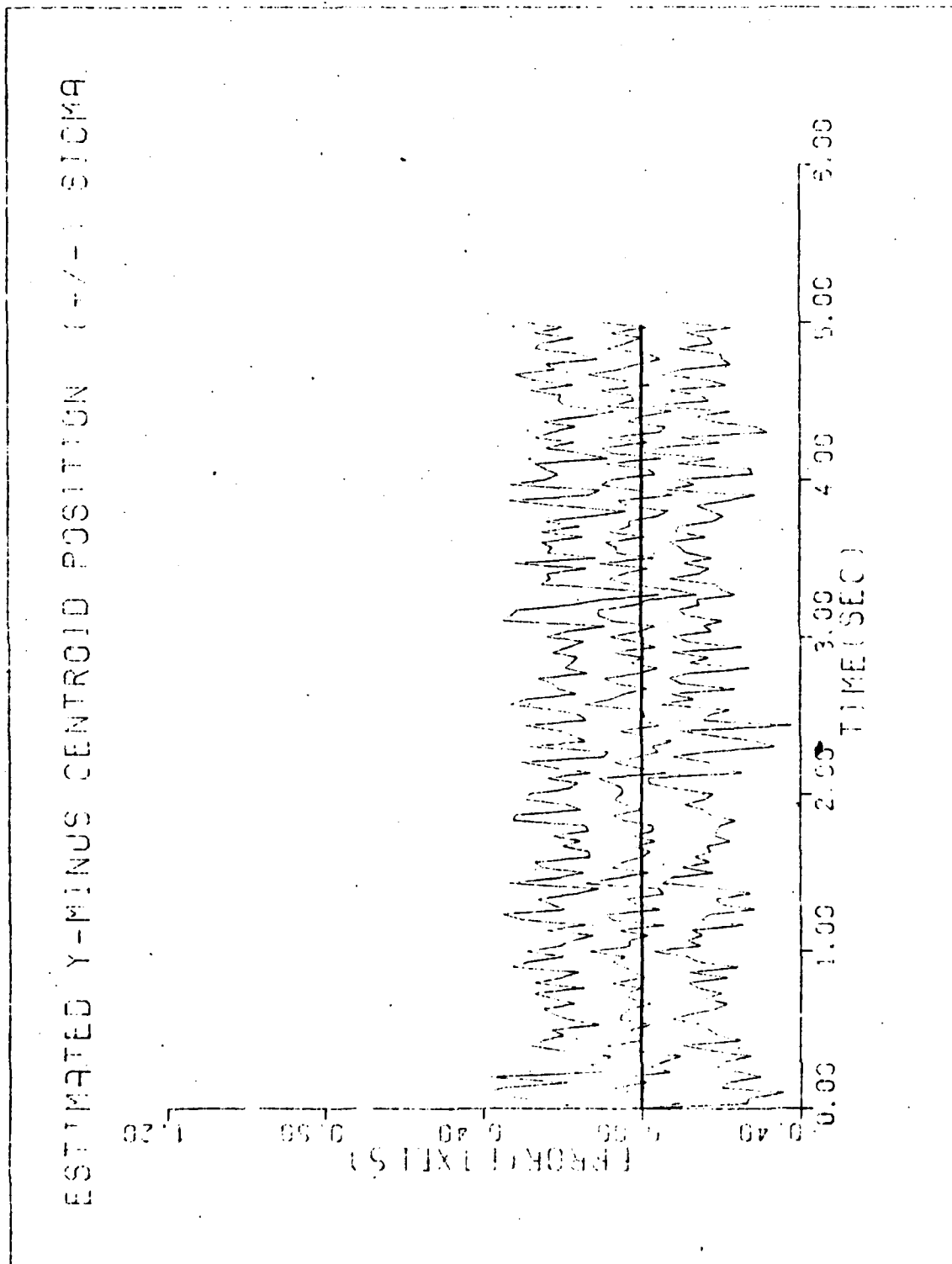


Figure C-16h. Case 16

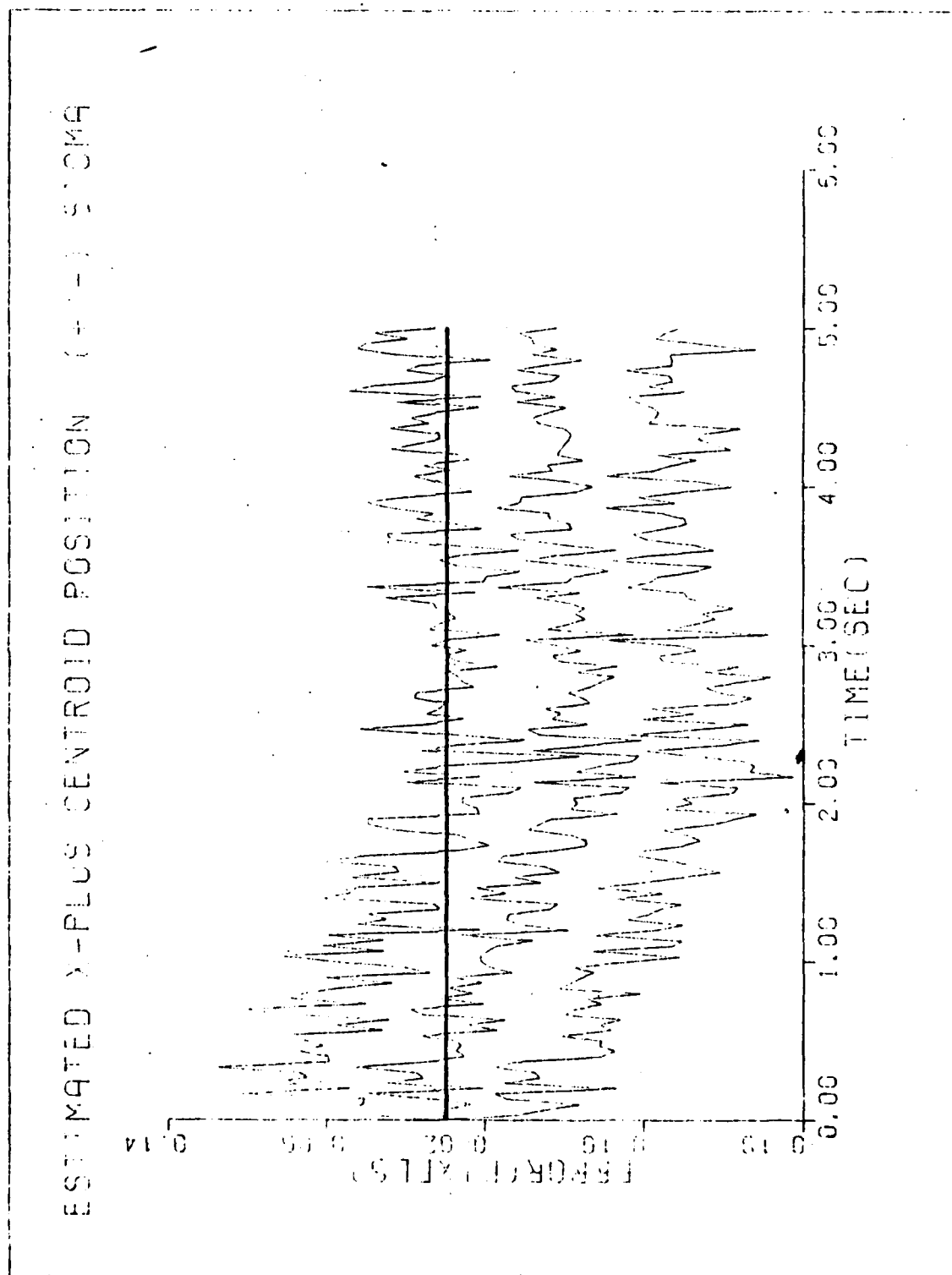


Figure C-161. Case 16

ESTIMATED Y-PLUS CENTROID POSITION (4-1) 5 CMG

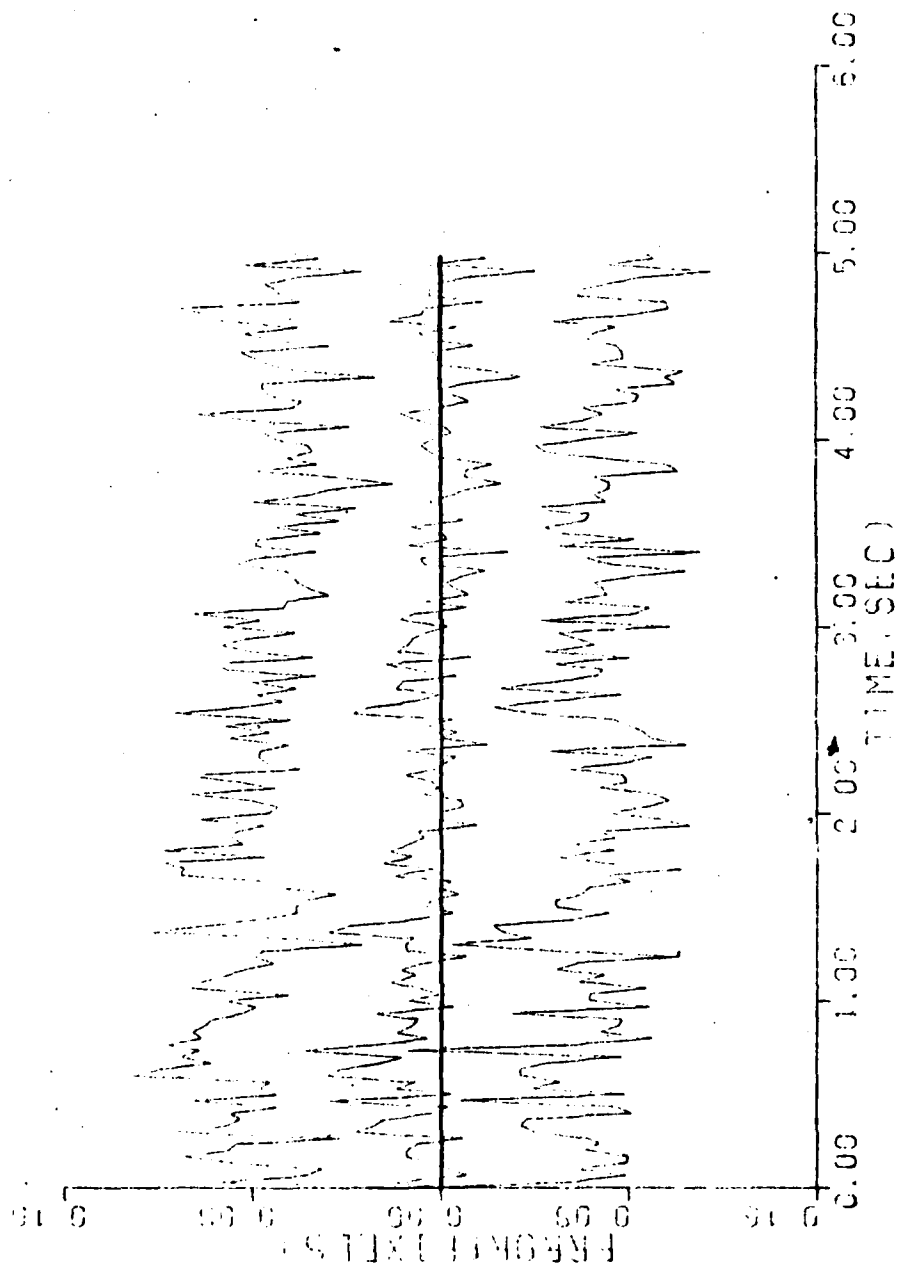


Figure C-16-j. Case 16

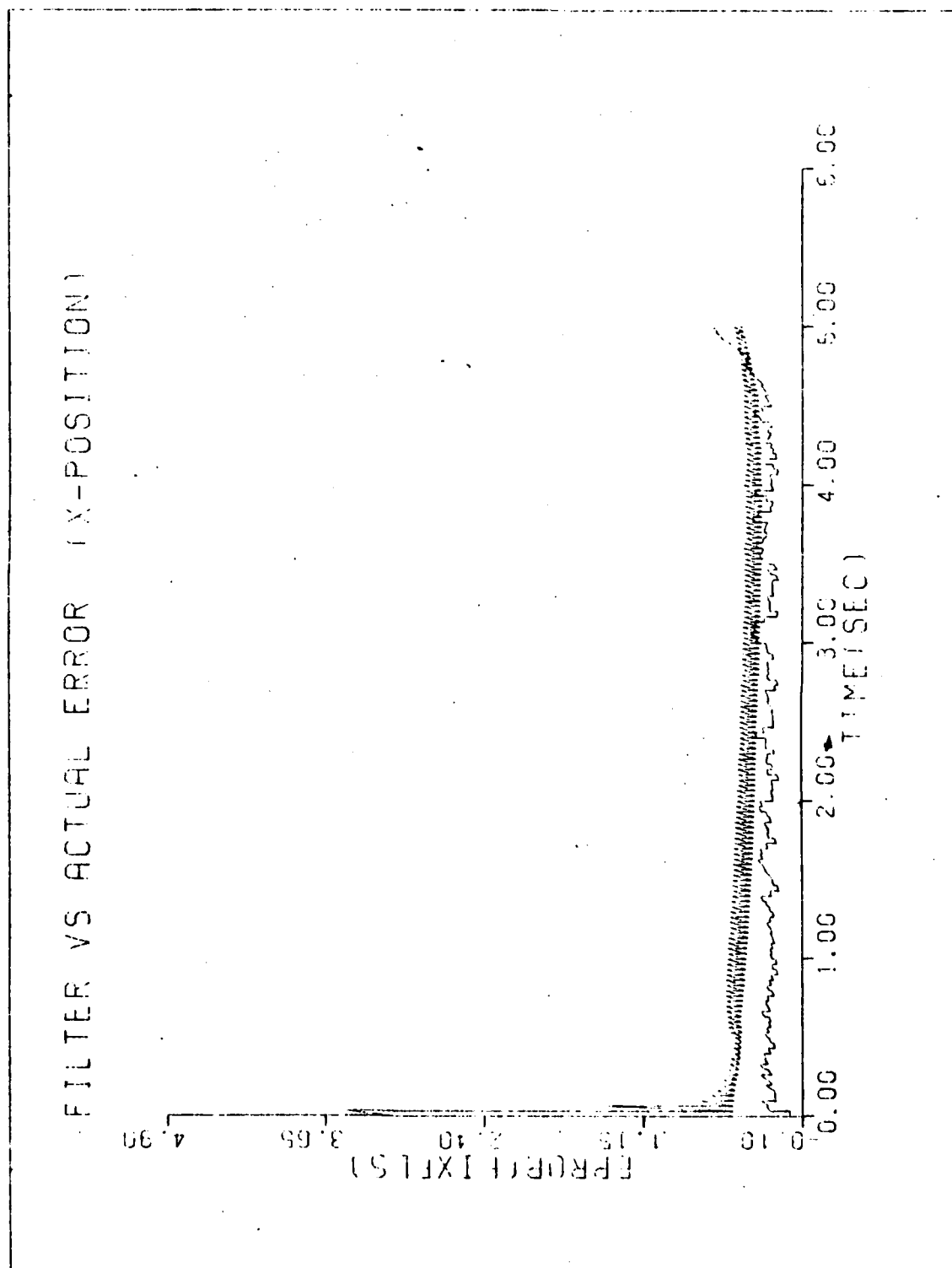


Figure C-17a. Case 17

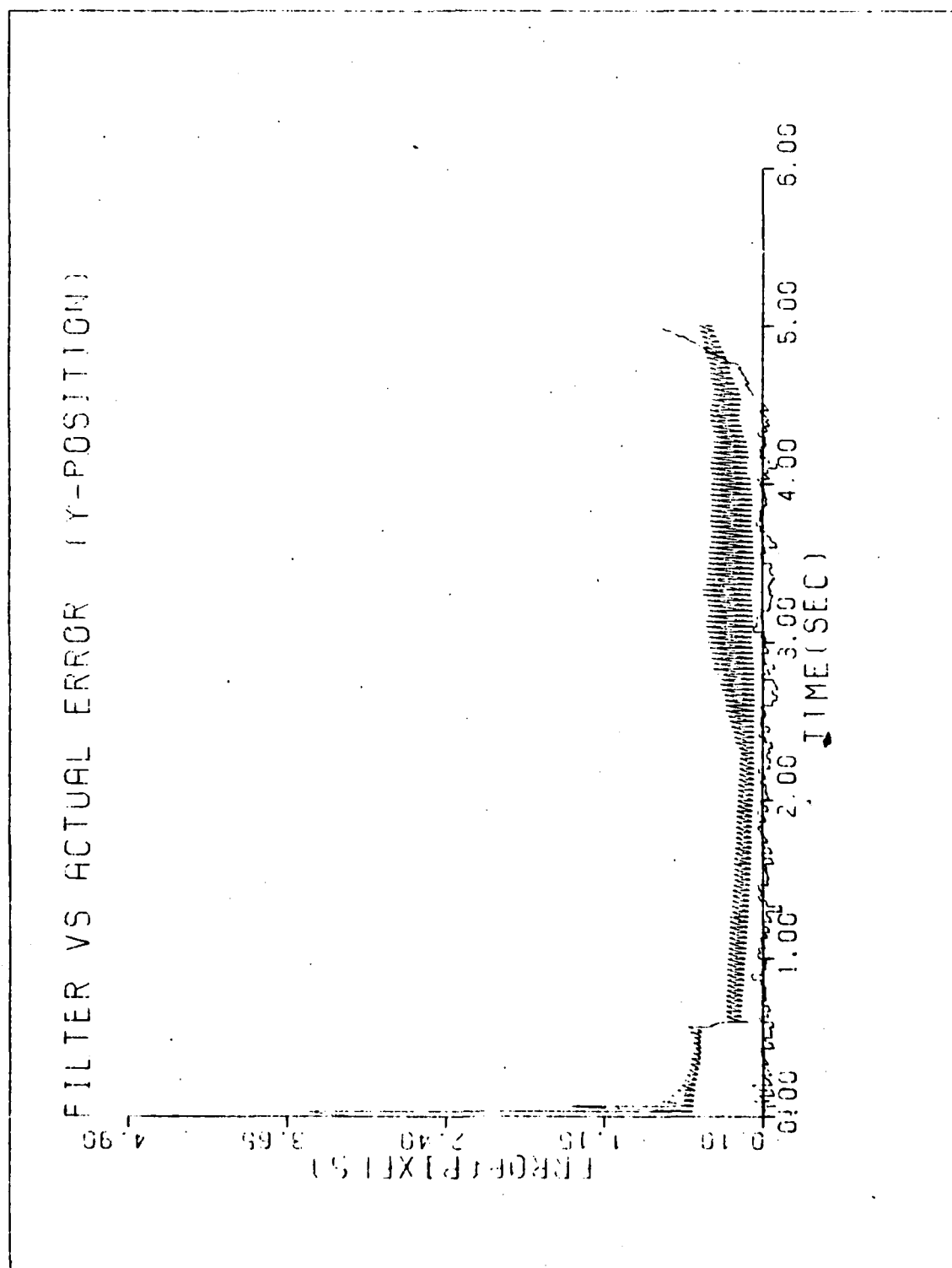


Figure C-17b. Case 17

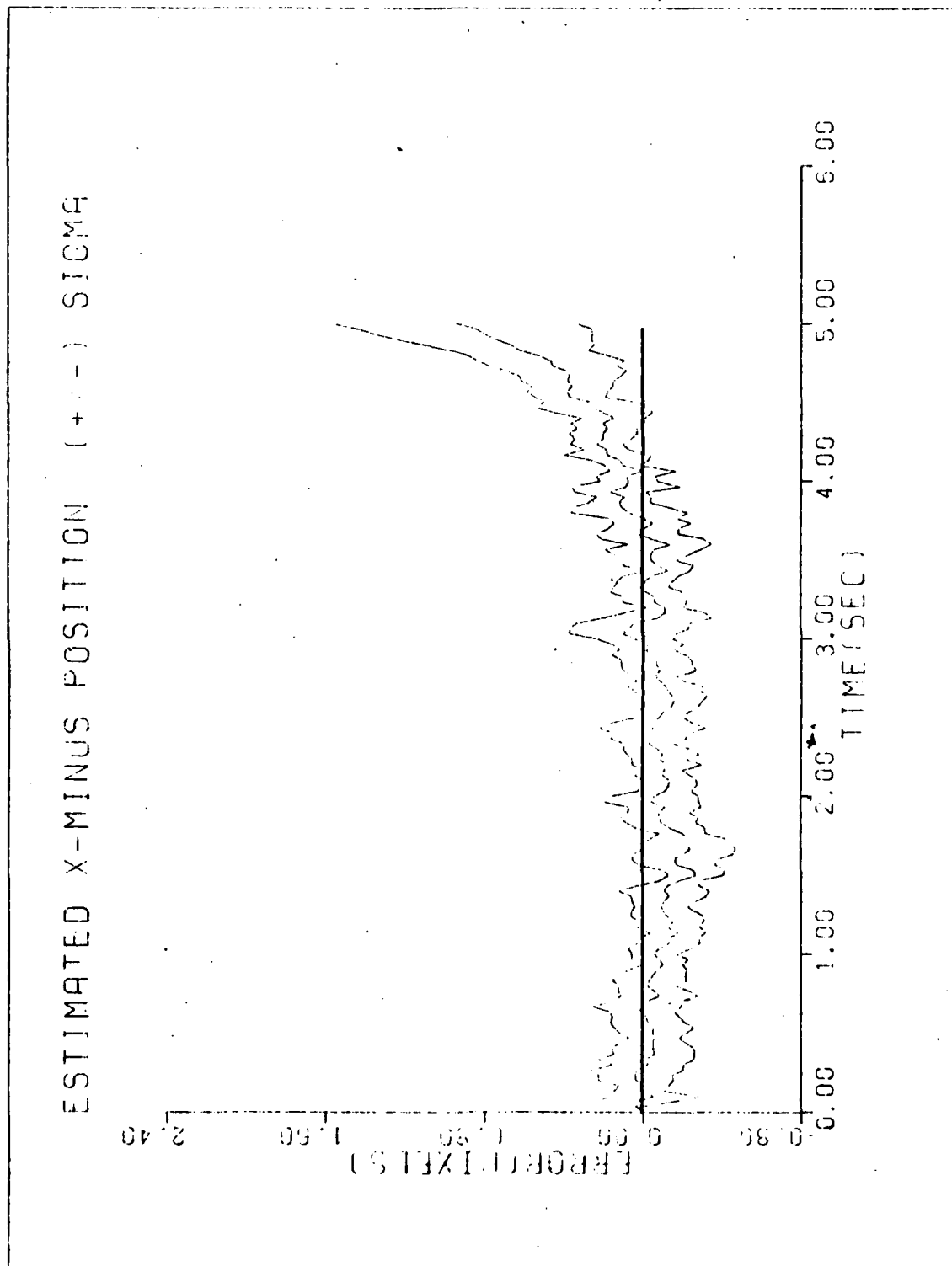


Figure C-17c. Case 17

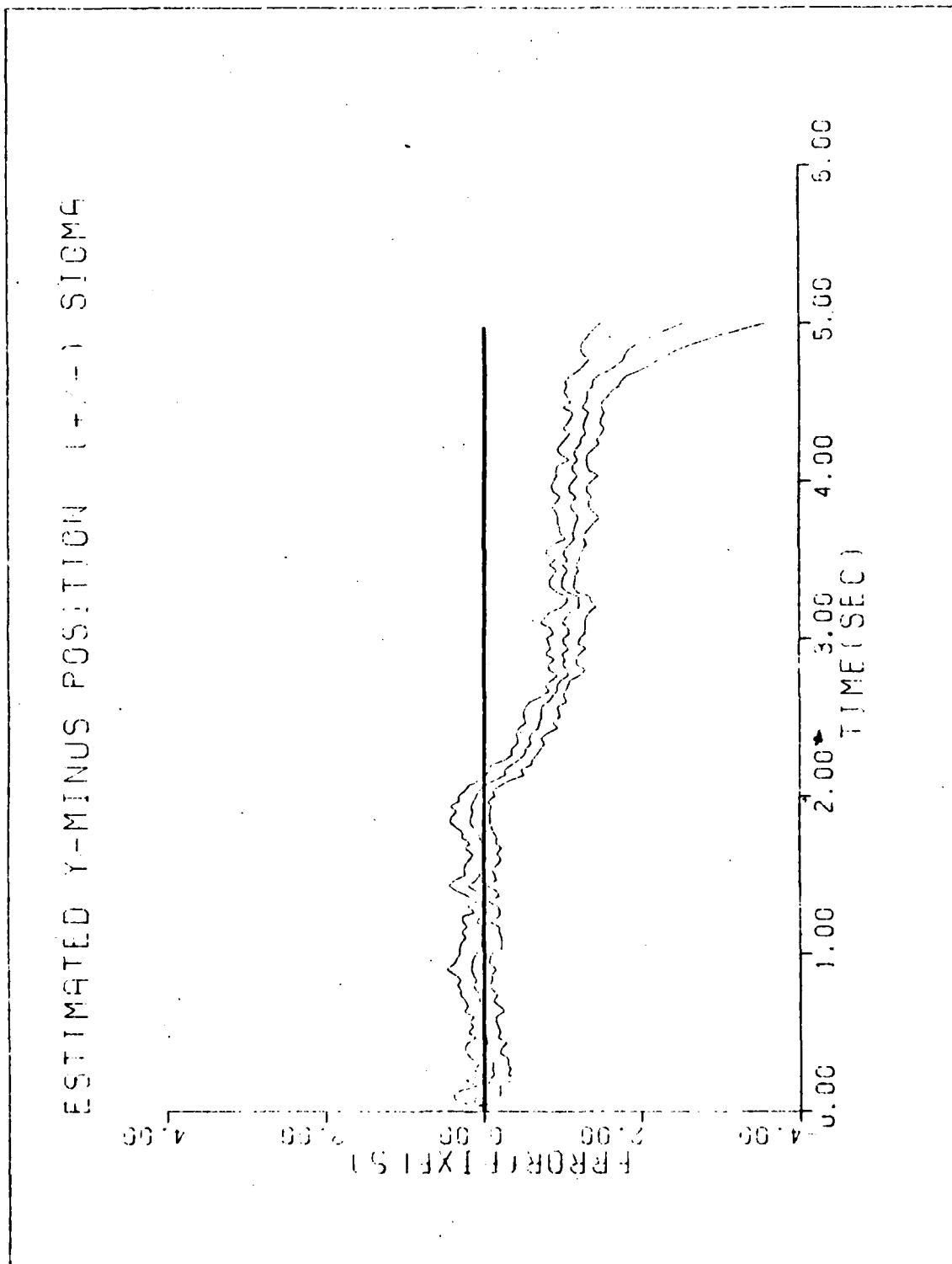


Figure C-17d. Case 17

6

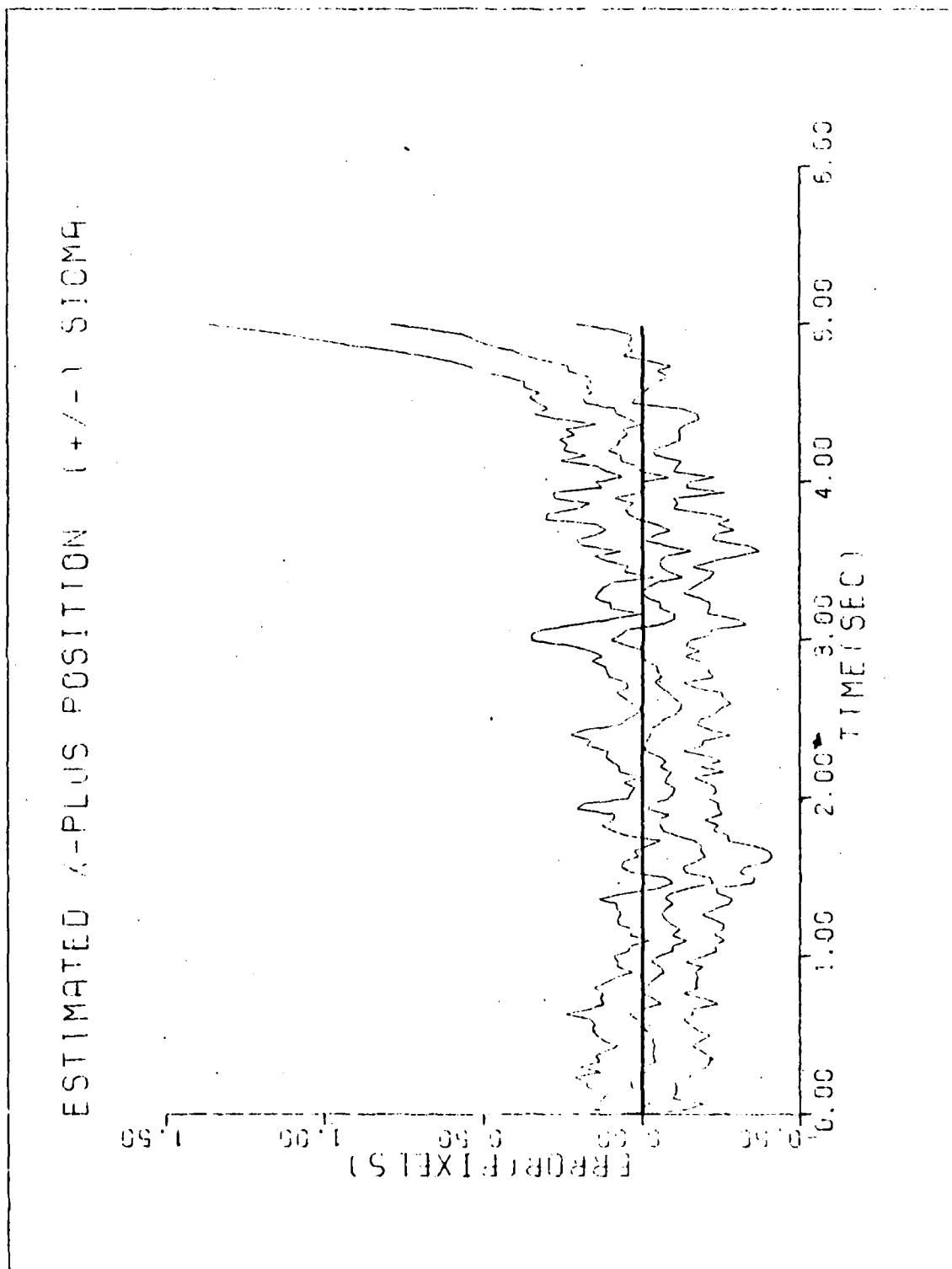


Figure C-17e. Case 17

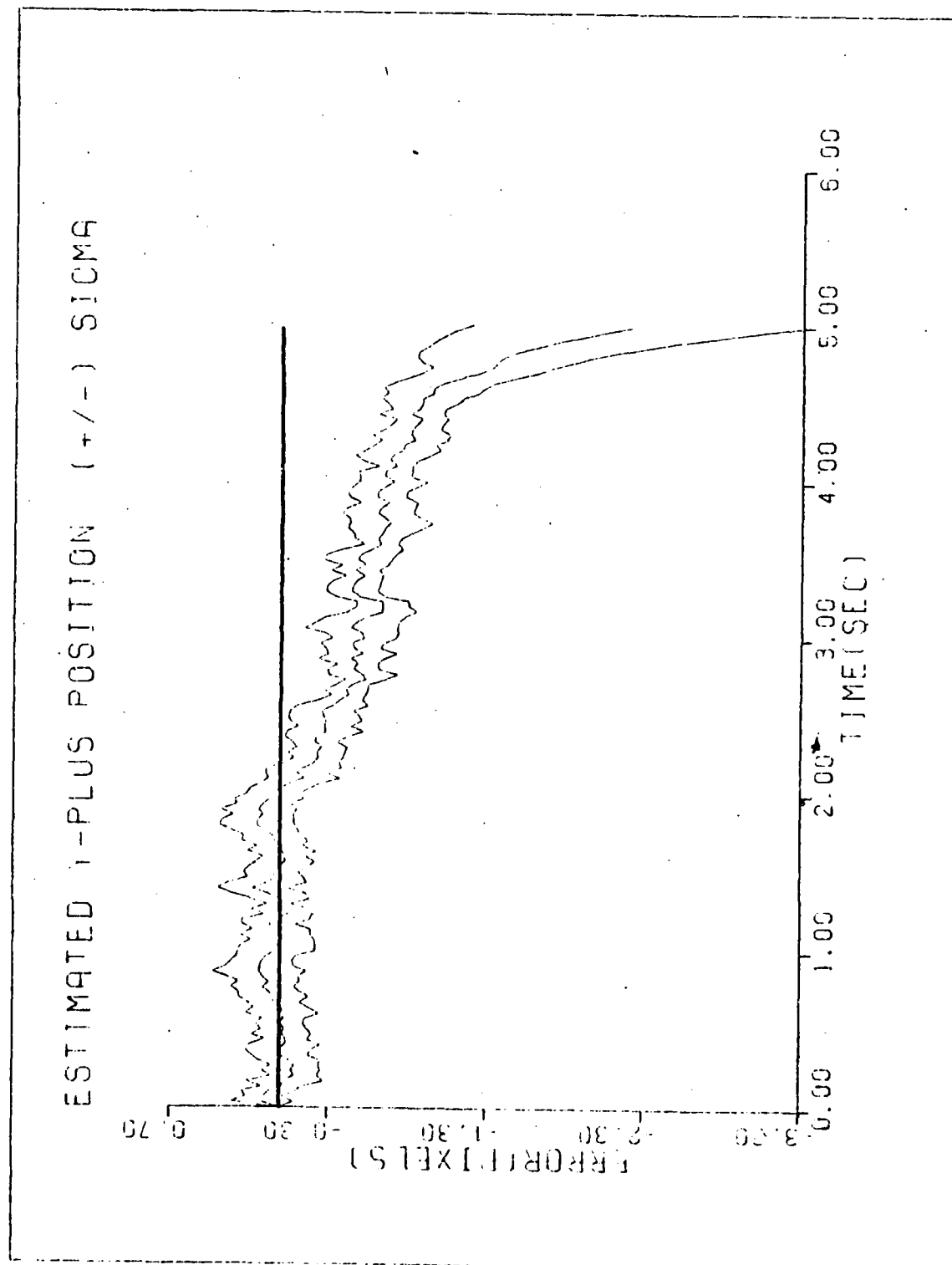


Figure C-17f. Case 17

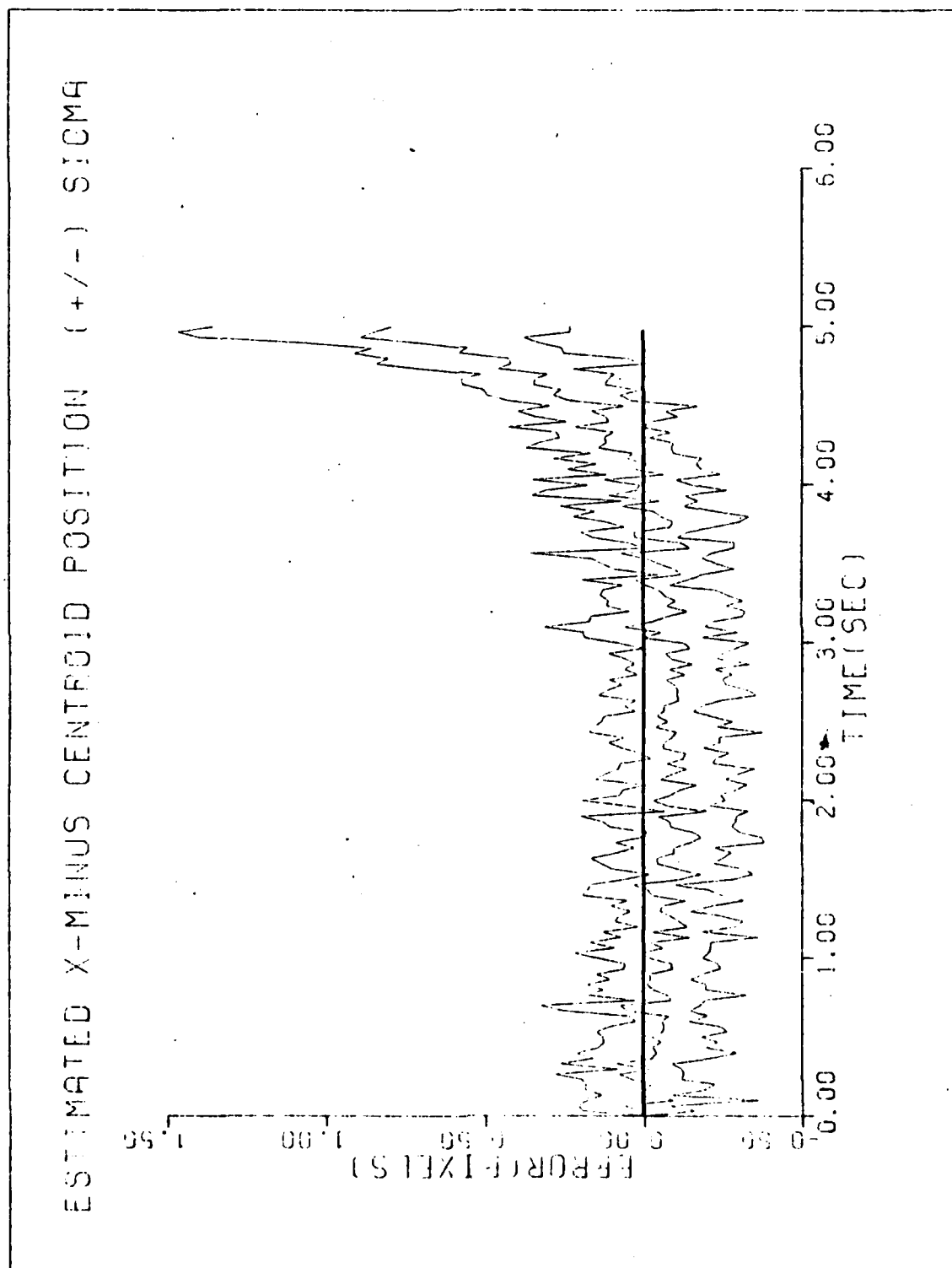


Figure C-17g. Case 17

ESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA

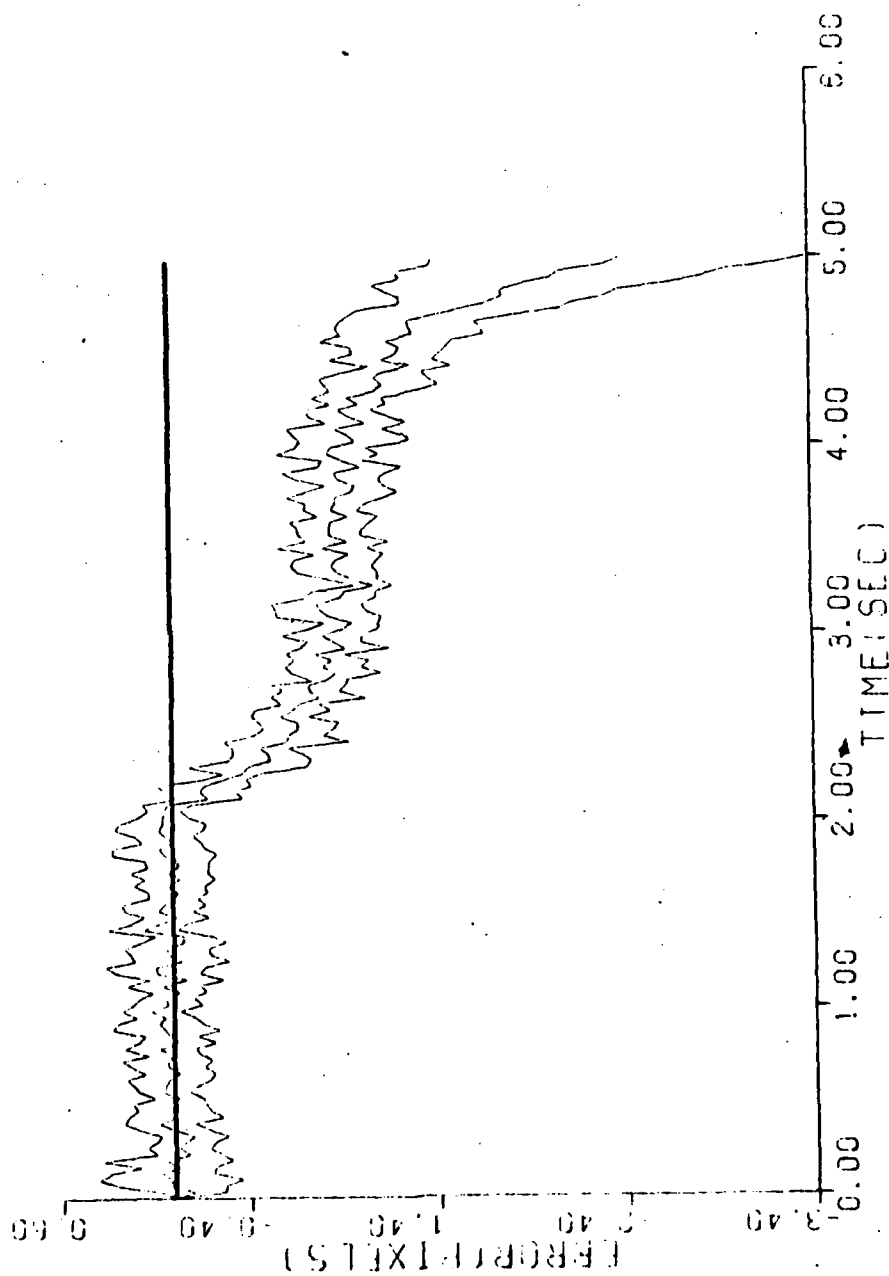


Figure C-17h. Case 17

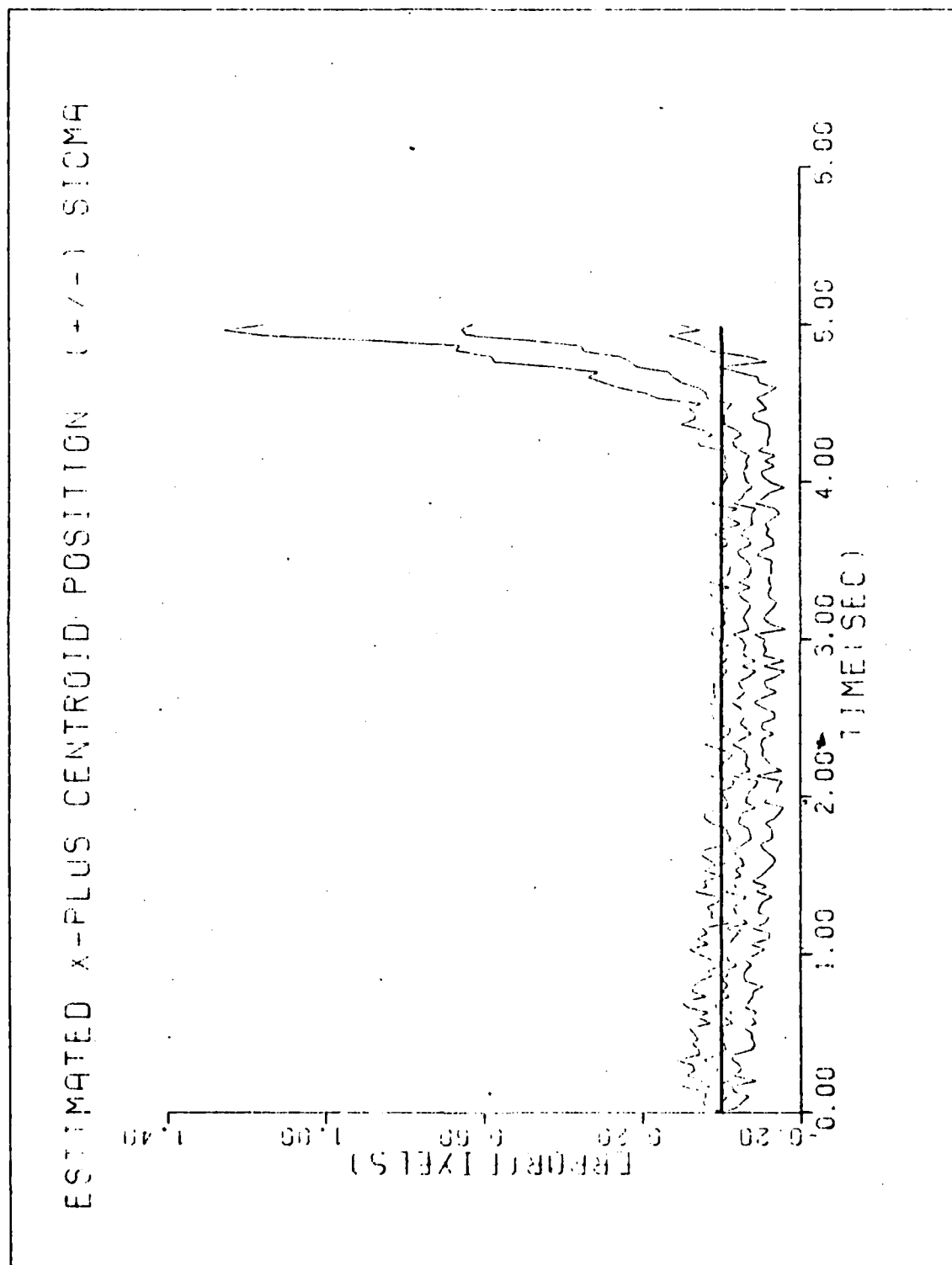


Figure C-17i. Case 17

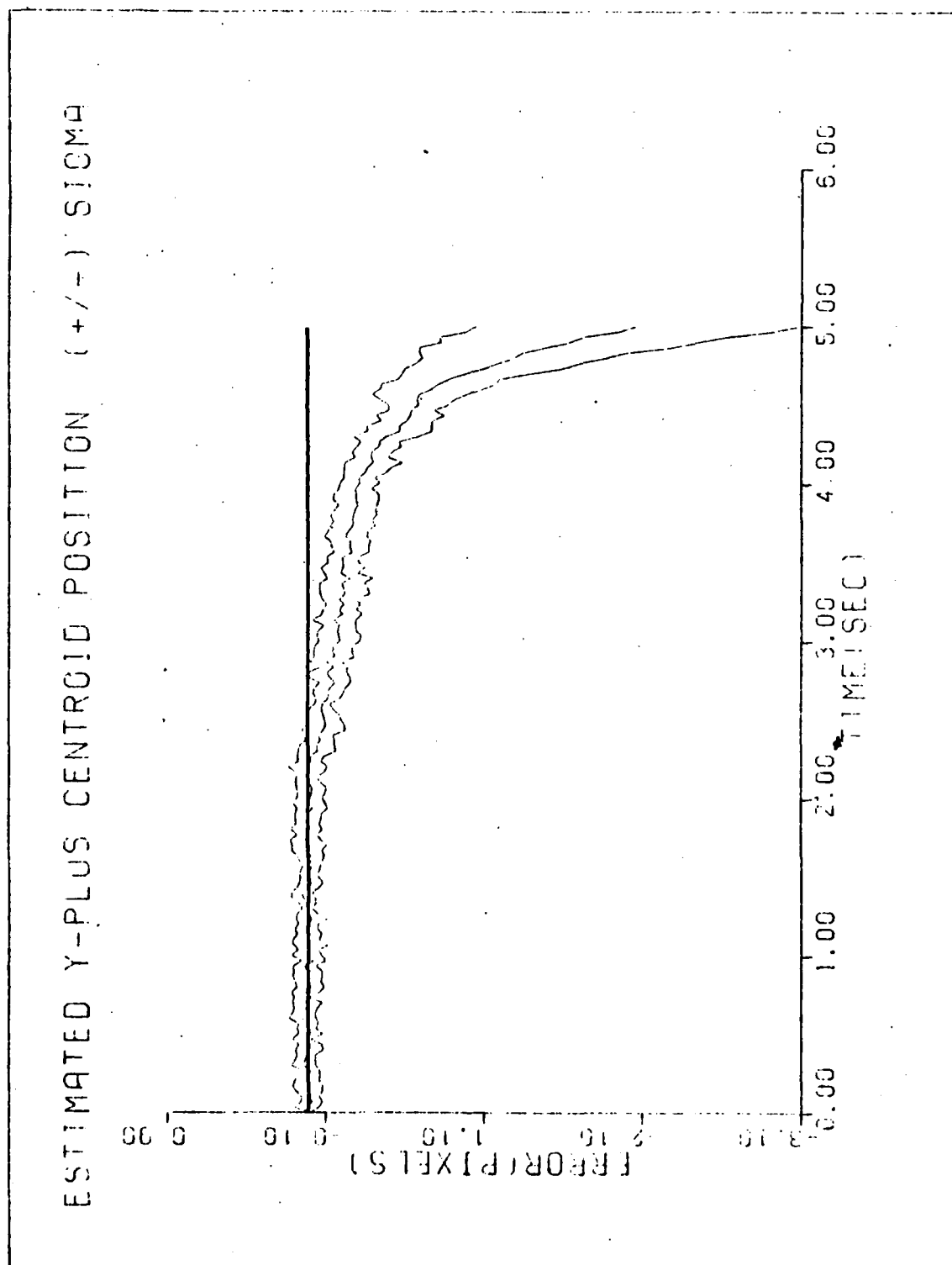


Figure C-17j. Case 17

The performance plots for this case were similar to the plots shown for case 12, thus they are omitted.

Figure C-18. Case 18

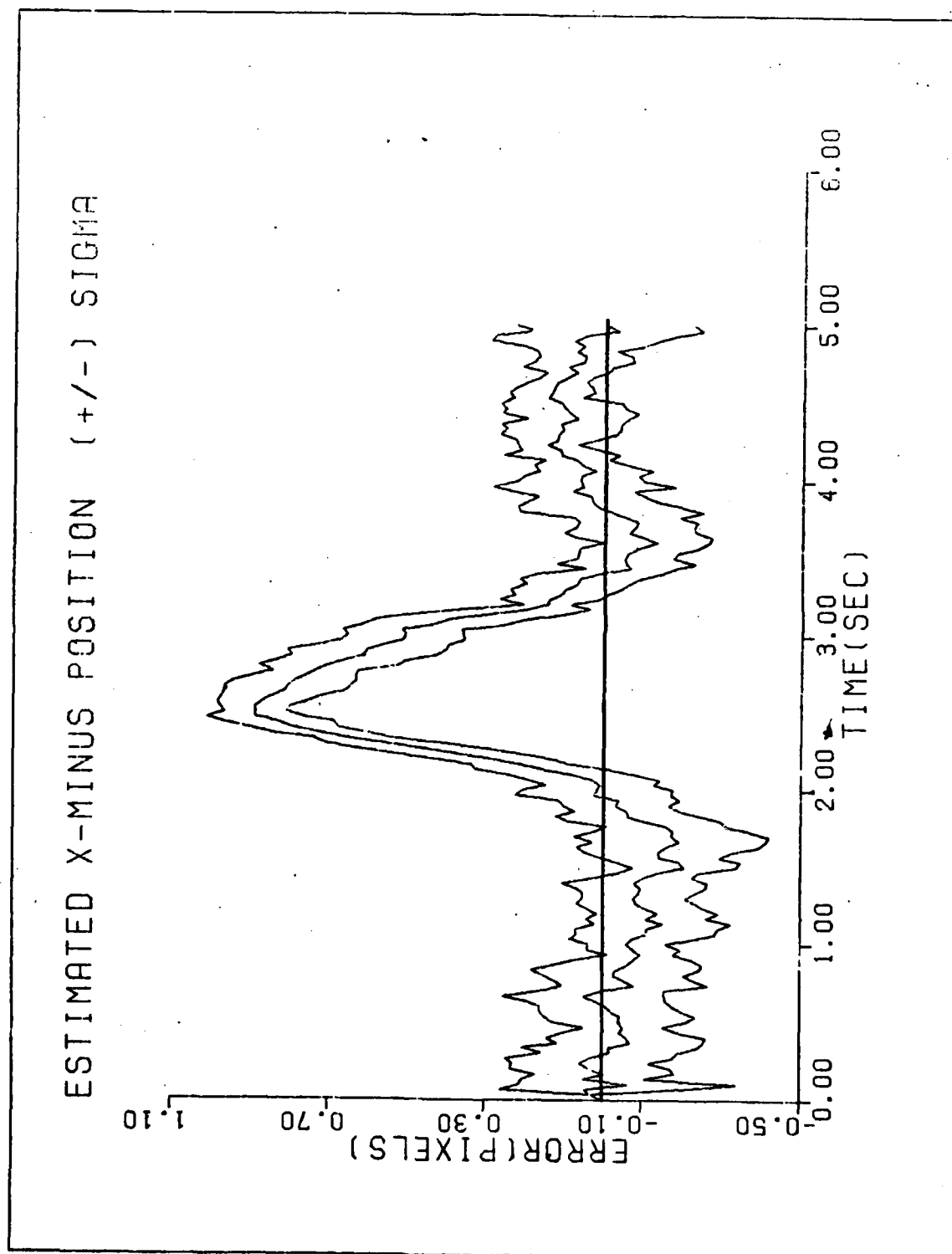


Figure C-19a. Case 19

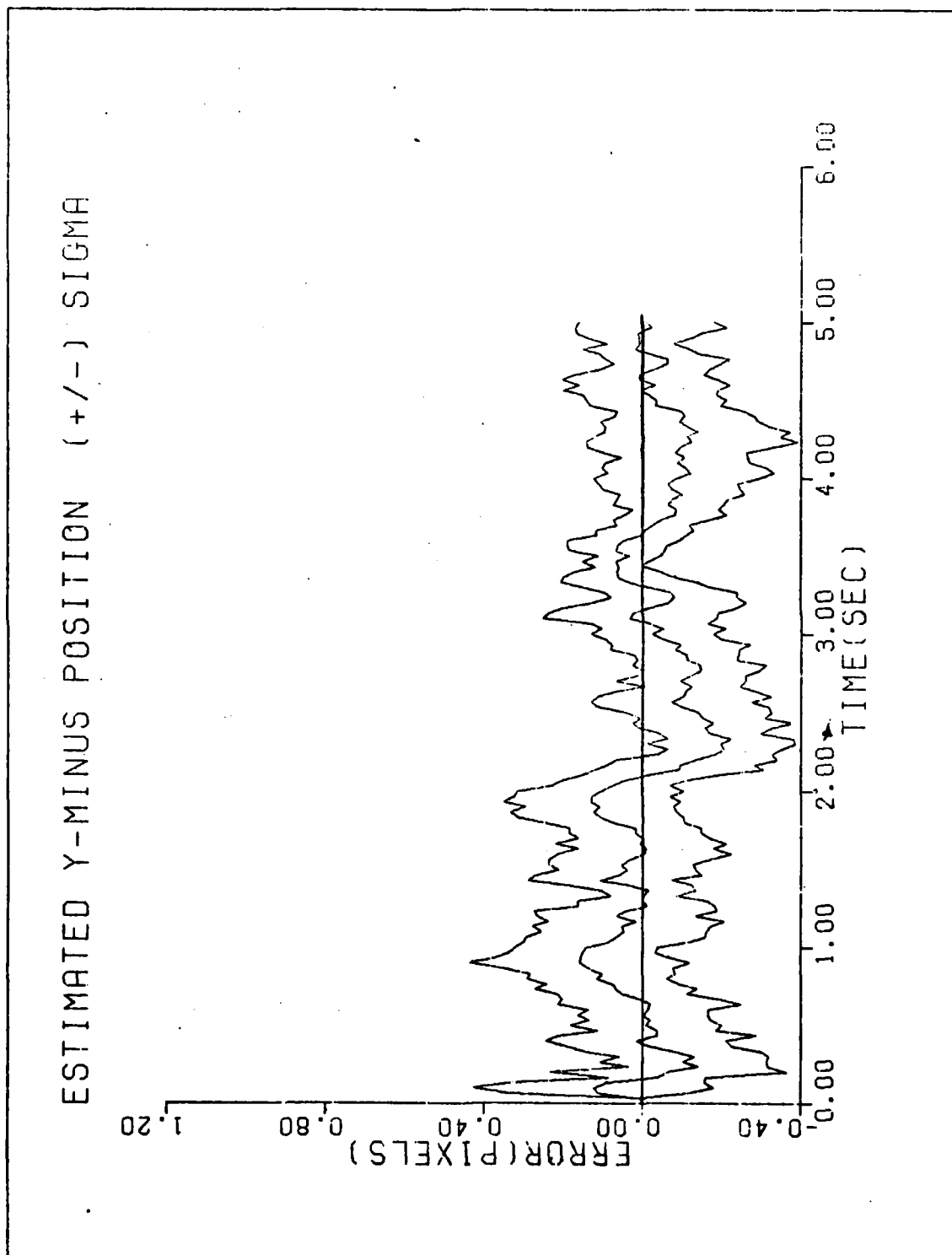


Figure C-19b. Case 19

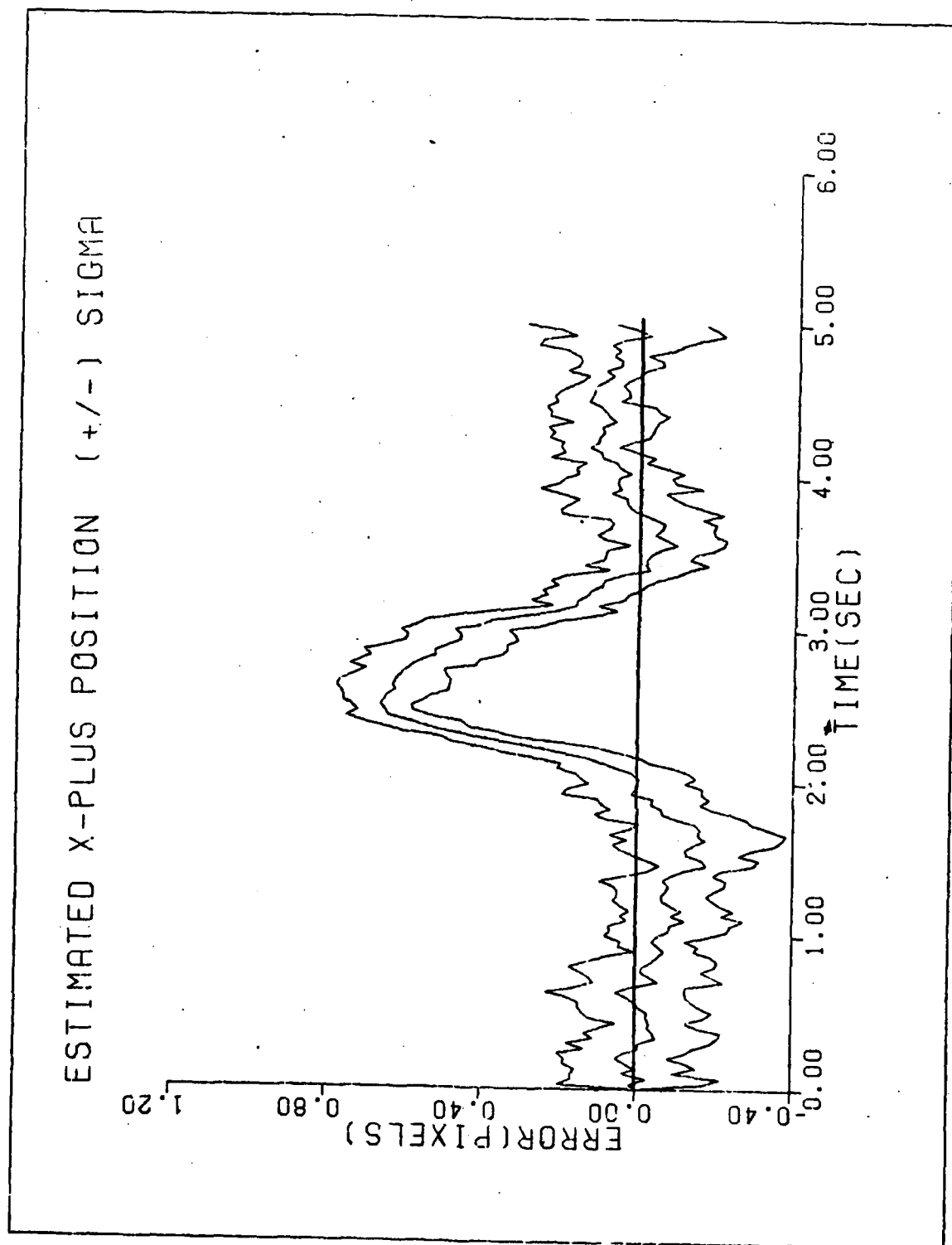


Figure C-19c. Case 19

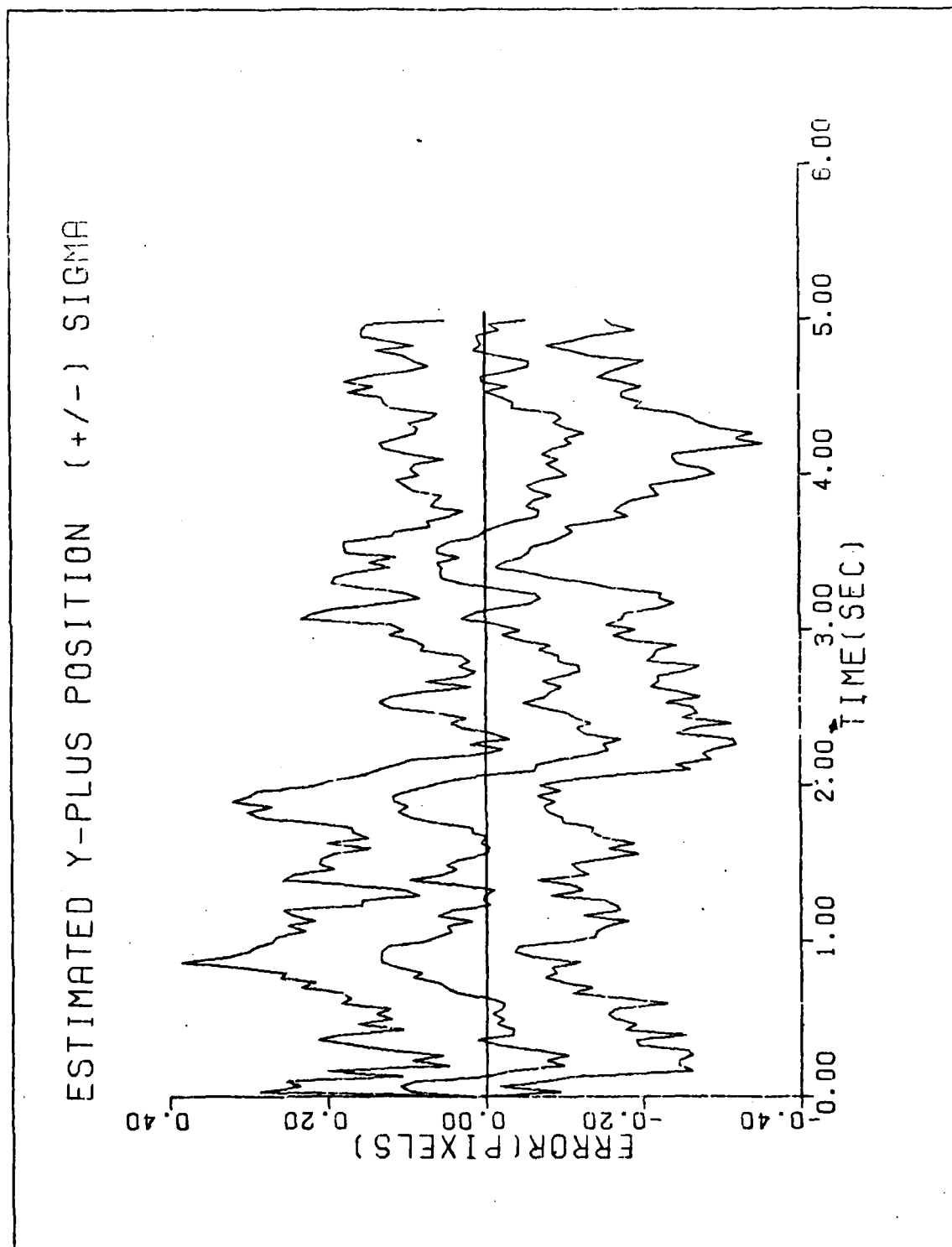


Figure C-19d. Case 19

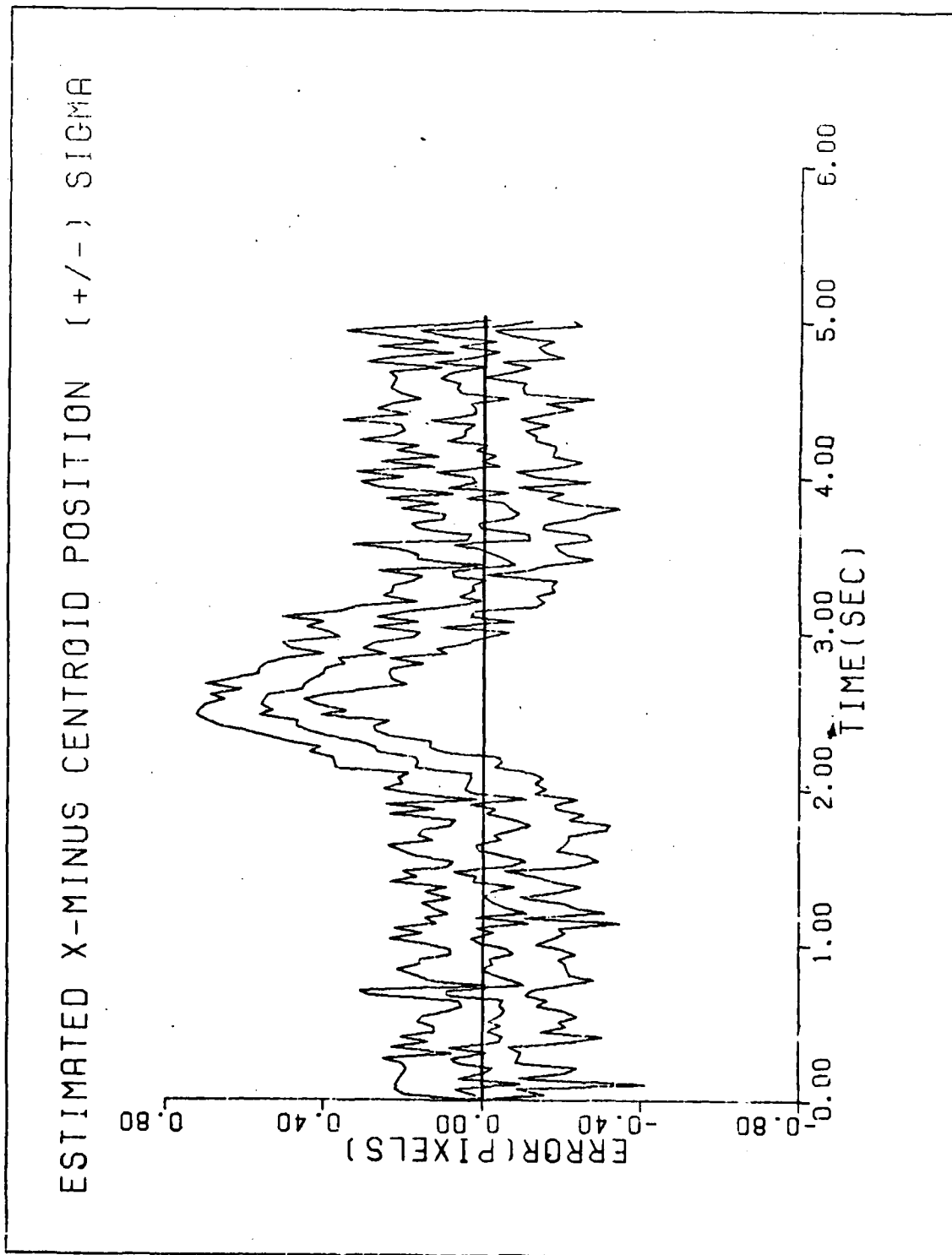


Figure C-19e. Case 19

AD-A124 884

ENHANCED TRACKING OF AIRBORNE TARGETS USING A
CORRELATOR/KALMAN FILTER(U) AIR FORCE INST OF TECH
WRIGHT-PATTERSON AFB OH SCHOOL OF ENGINEERING
P P HILLNER DEC 82 AFIT/GE/EE/82D-58

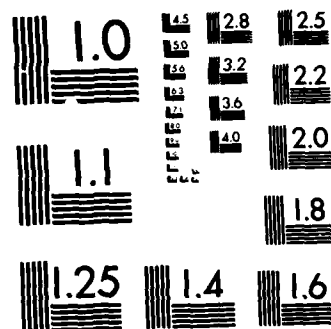
4/4

UNCLASSIFIED

F/G 12/1

NL





MICROCOPY RESOLUTION TEST CHART
NATIONAL BUREAU OF STANDARDS-1963-A

ESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA

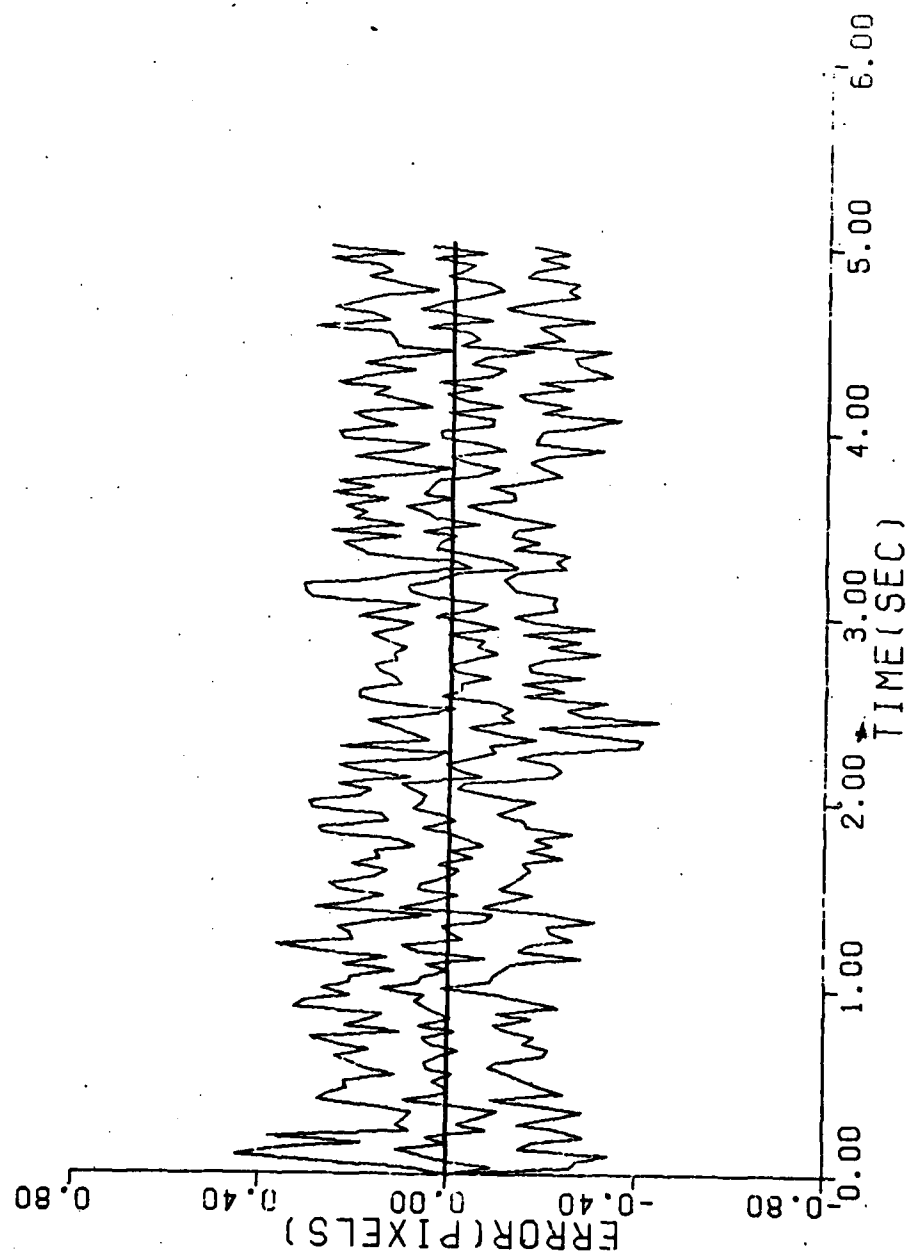


Figure C-19f. Case 19

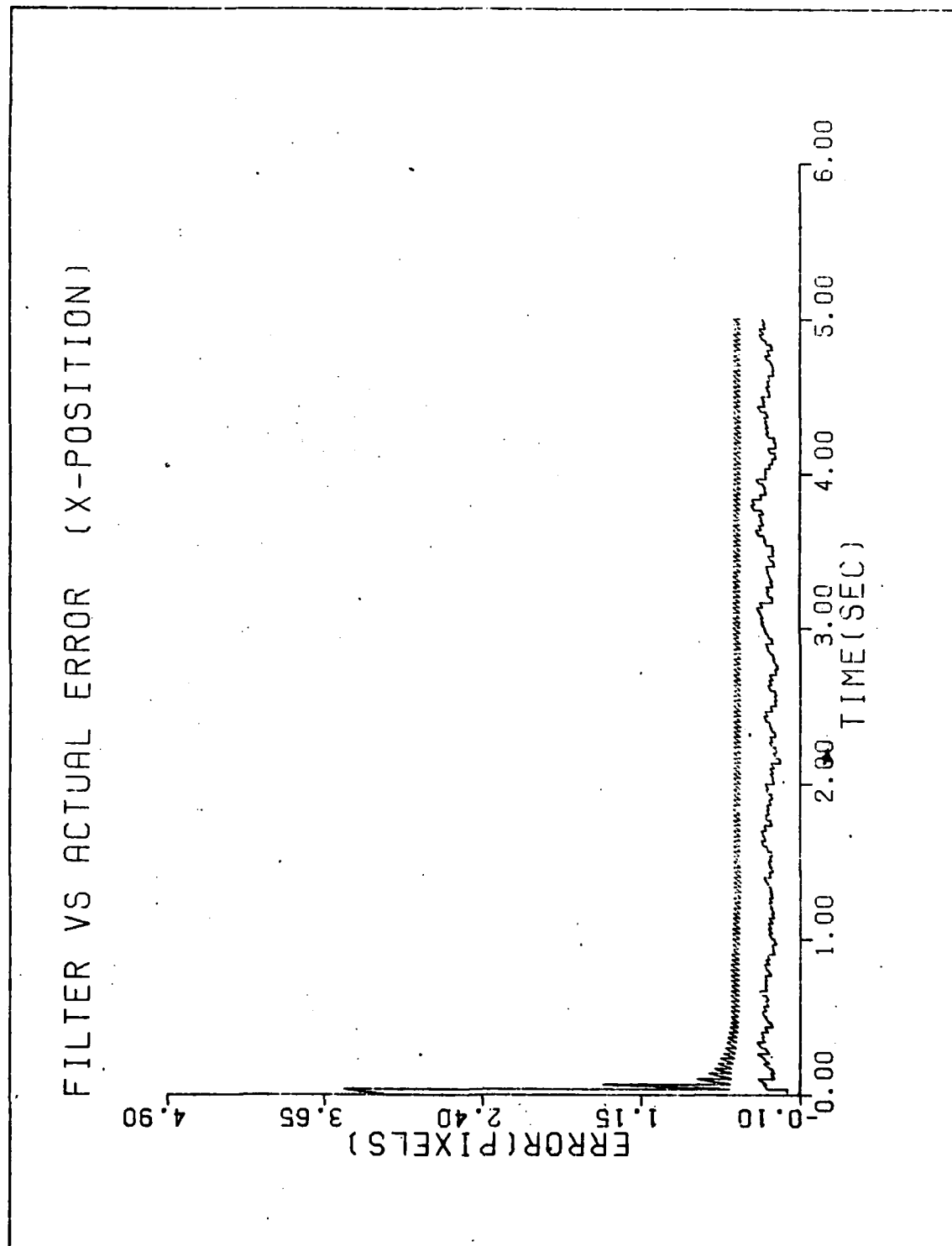


Figure C-20a. Case 20

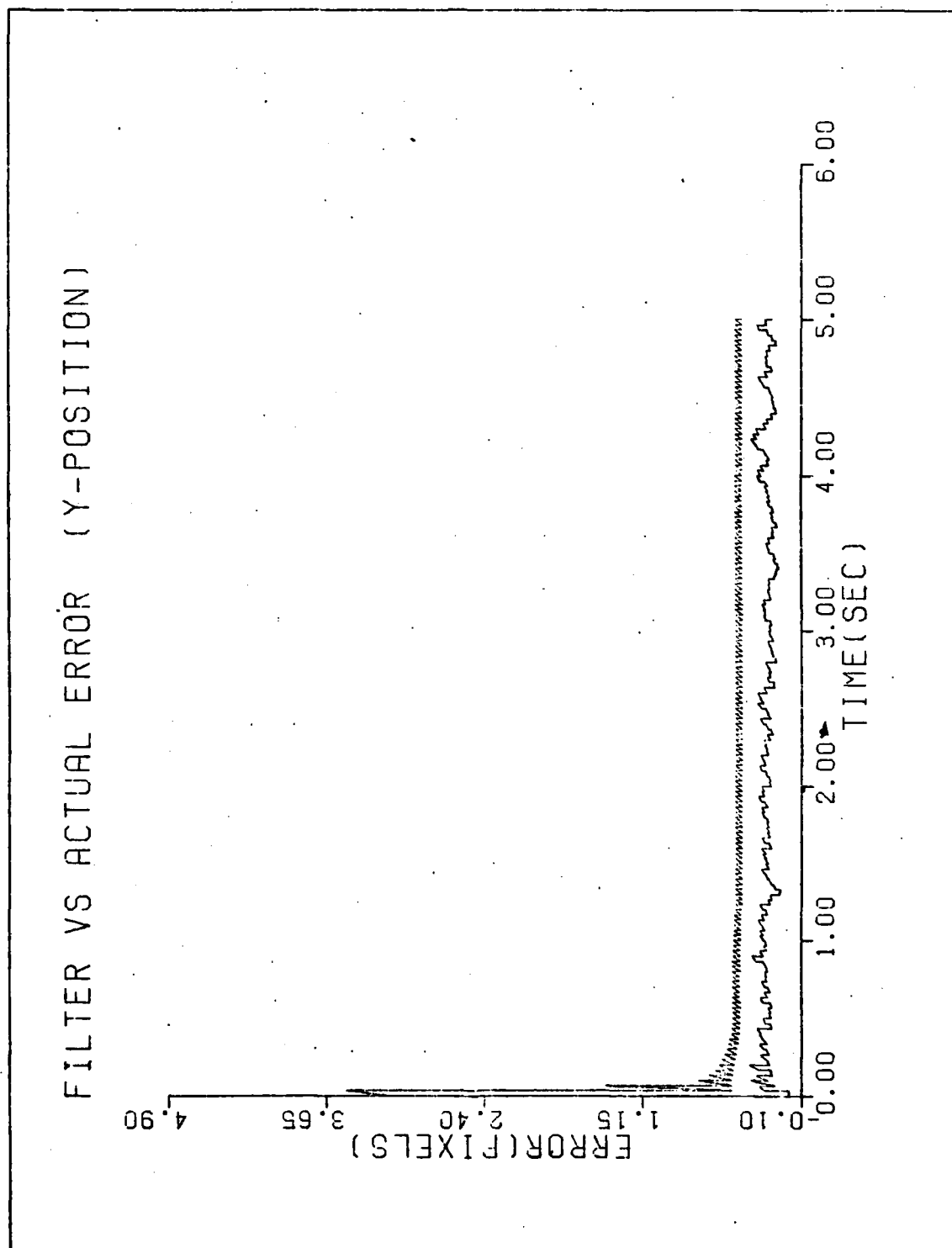


Figure C-20b. Case 20

ESTIMATED X-MINUS POSITION (+/-) SIGMA

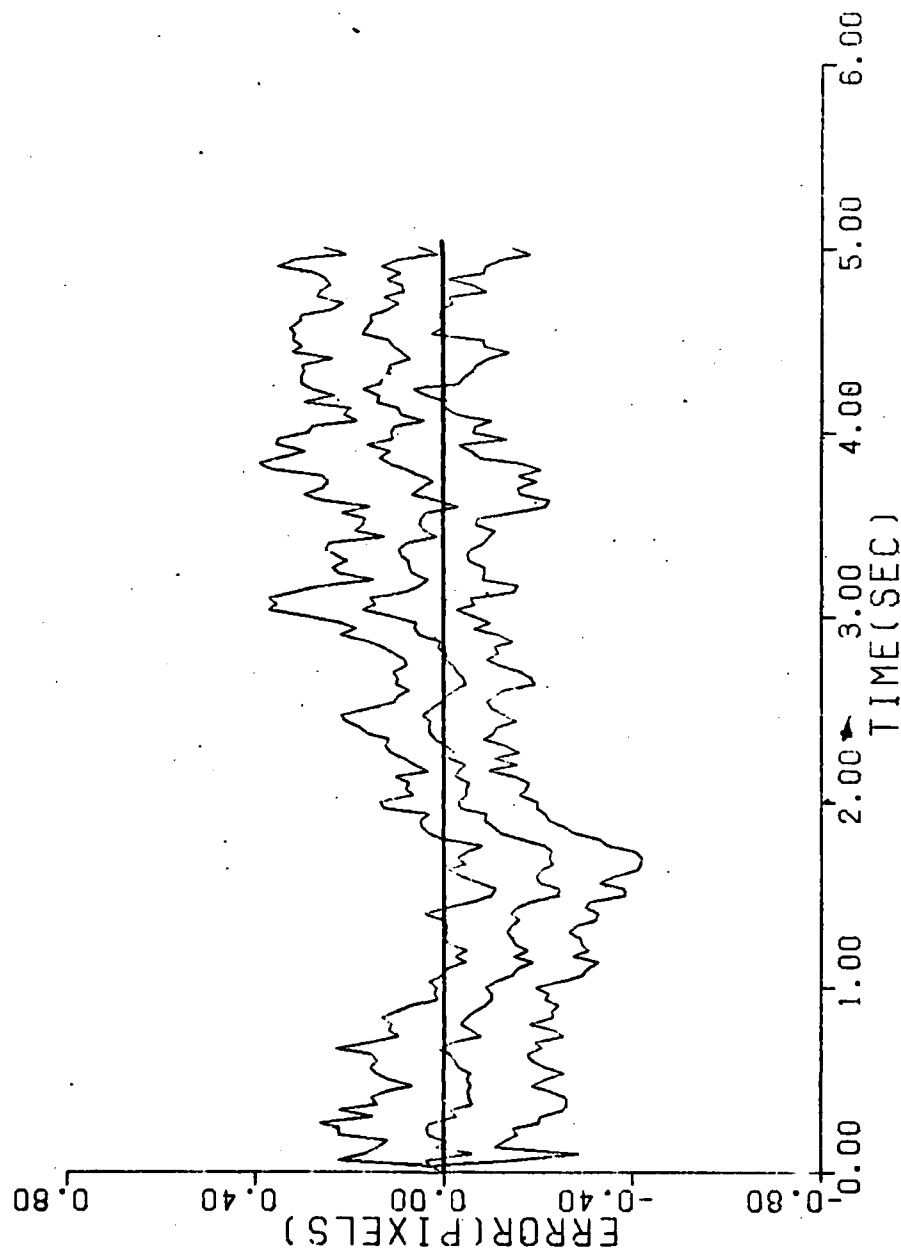


Figure C-20c. Case 20

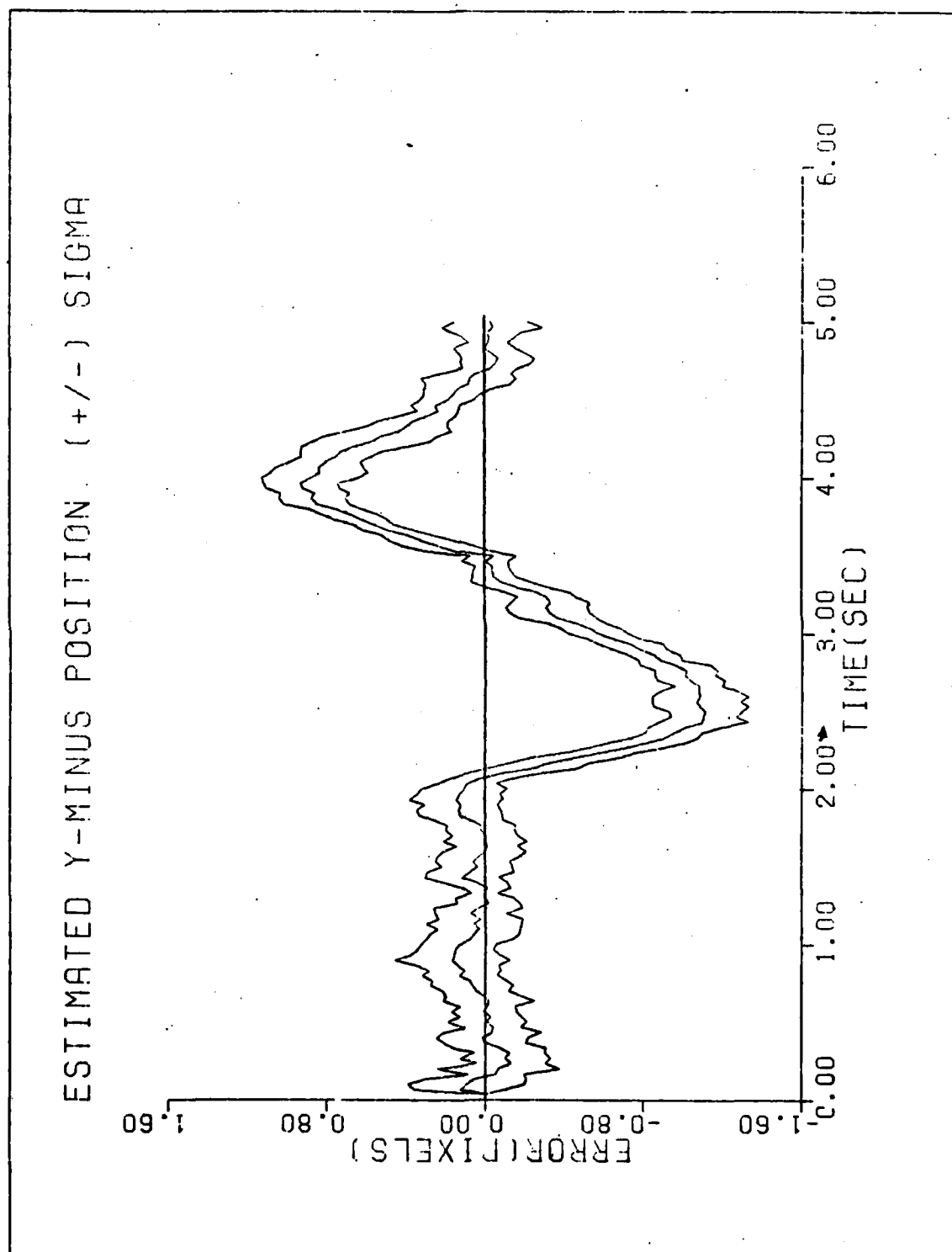


Figure C-20d. Case 20

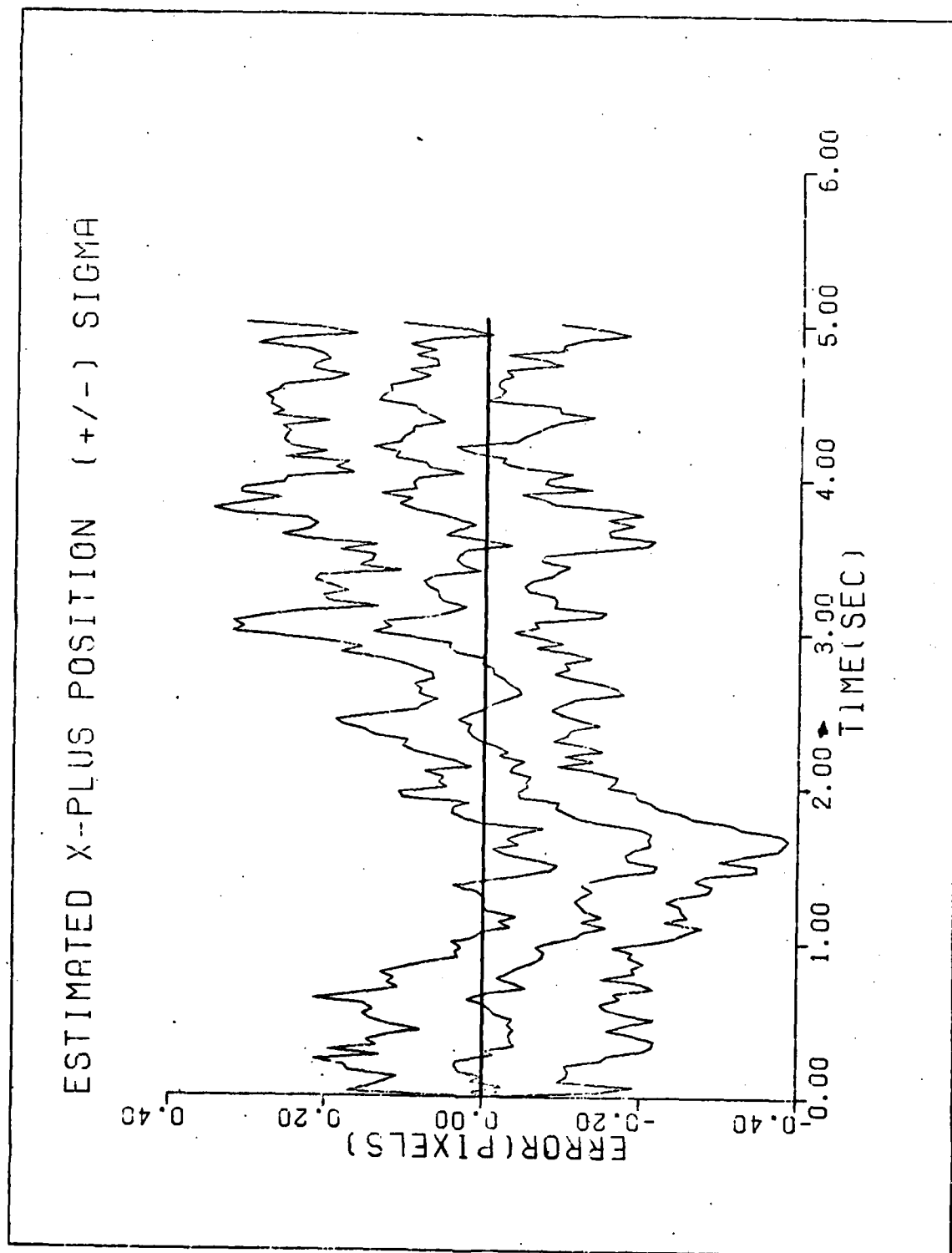


Figure C-20e. Case 20

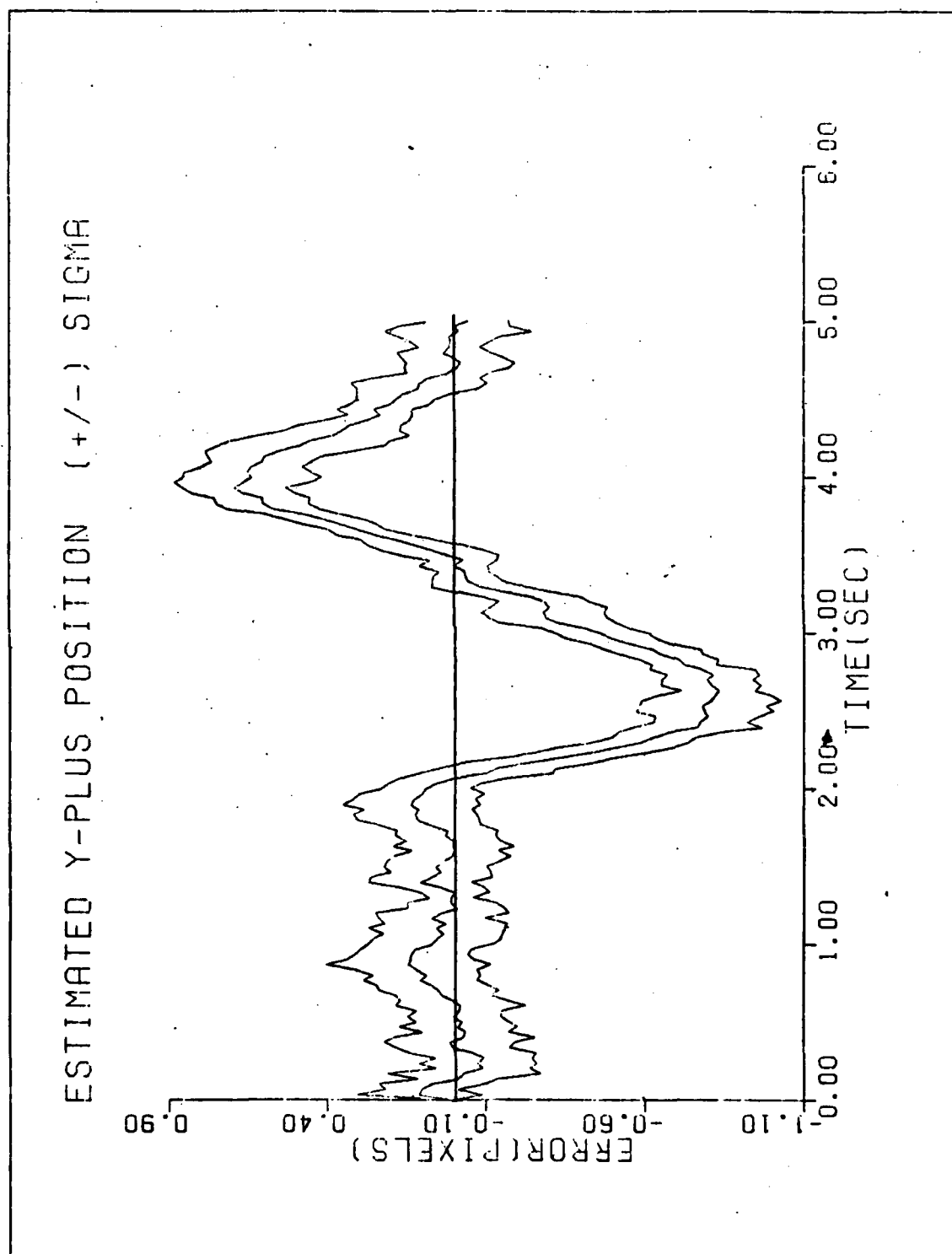


Figure C-20f. Case 20

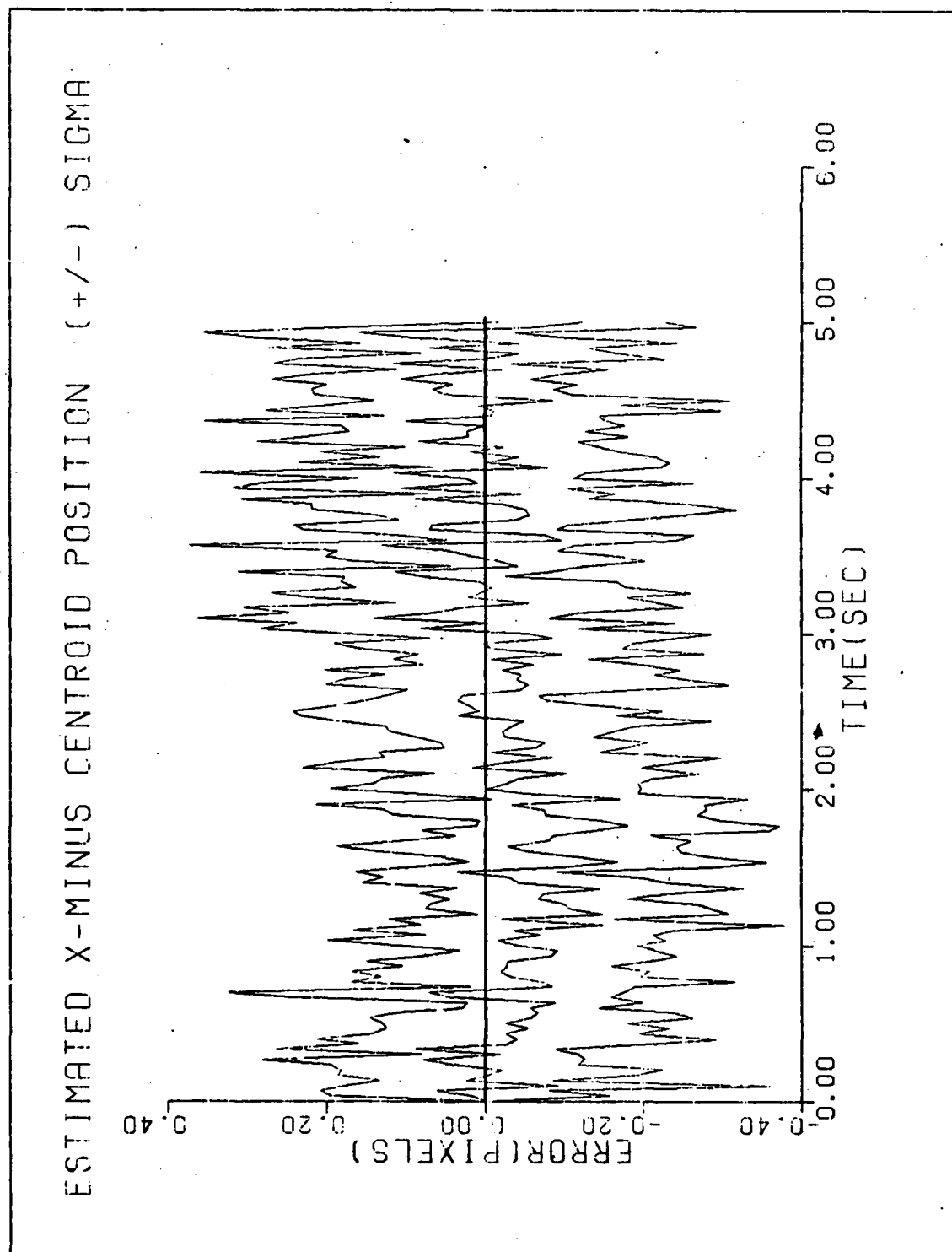


Figure C-20g. Case 20

ESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA

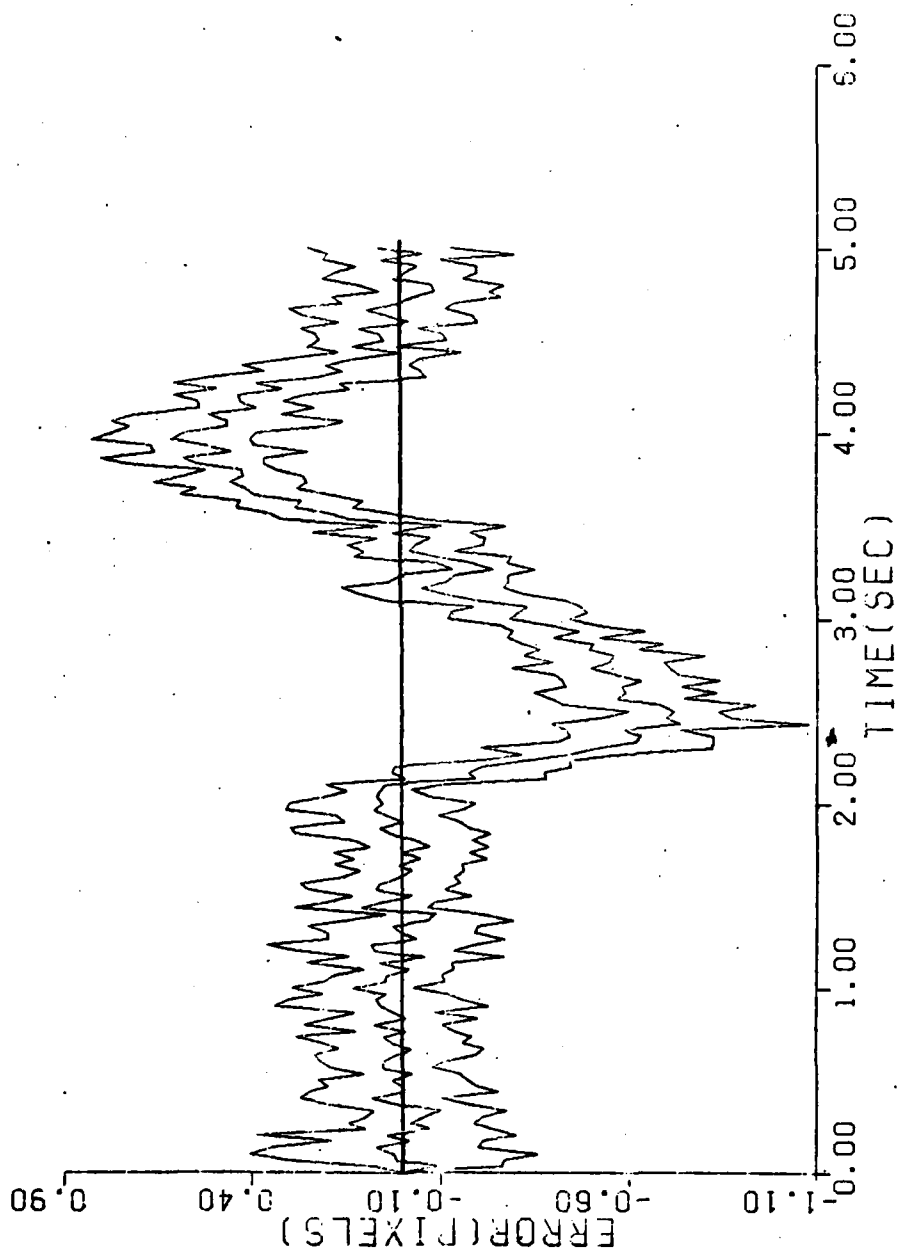


Figure C-20h. Case 20

Appendix D

Computer Software (Correlator/Kalman Filter)

This appendix contains the Fortran source code for the implementation of the algorithm of Figure 1. The program also contains the truth model described in Chapter 2 where the trajectory can be generated either with the model internal to this program or using the trajectory program given in Appendix E. (Note: For the multiple hot spot model the program given in Appendix E is used). Finally, this program contains the routines to calculate and plot the tracking errors of the algorithm which are presented in Chapter 5 and Appendix C. The program was written for use on the CDC Fortran IV compiler.

```

PROGRAM MAIN (INPUT, OUTPUT, TAPE5=INPUT, TAPE6=OUTPUT, TAPE1,
1  DEBUG=OUTPUT, TAPE4)
  REAL IMAX(3),S(12),XMAX(3),YMAX(3),Z(64,64)
  REAL Z1(7)
  REAL RFIL(2, )
  REAL XT(8,1),PHIT(8,8),ODPROT(8,8),H(2, ),YT(2,1)
  REAL W(64),V(64),UC(576)
  INTEGER IN(2),NFRAMES,NUMHS,TPAGEN,STRUN,NIM1
  COMPLEX DATA(24,24),WORK(50),SAVE(24,24),DX(24,24),DY(24,24)
  COMPLEX SDA A(24,24)
  REAL UT(2,1),HD(1,2)
  REAL FILPL(400),ACTPL(400),FILPY(400),ACTPY(400)
  REAL TIMPL(400),TIME(200)
  REAL PTA(400),PTB(400),PTC(400),PTD(400)
  REAL SIGMS,ADP=0
  REAL UPD(8),VFPF(8,8)
  INTEGER QPRINT,ICHO
  REAL EXUT(2,200),HS1A(200),HS1B(200),HS2A(200),HS2B(200)
  REAL HS3A(200),HS3B(200),VMAXEX(200),RANEX(200),TIMEX(200)
  REAL COSW(200)
  INTEGER CORRL,IPL0T

```

THE FOLLOWING IS A LIST OF THE MAJOR PROGRAM SUBROUTINES
AND A BRIEF DESCRIPTION OF THE SUBROUTINES ROLE IN THE
ALGORITHM

CHOLY-IMPLEMENTS THE CHOLESKY SQUARE ROOT ROUTINE
CENTRO-PERFORMS THE CENTER OF MASS CALCULATION FOR THE CORRELATOR
CORREL-IMPLEMENTS THE CORRELATION ROUTINE FOR THE FFT AND
PHASE CORRELATION ROUTINES
CORL2-IMPLEMENTS THE DIRECT CORRELATION METHOD (2 AND 6
LEVEL QUANTIZATION)
FILPT-ARRANGES THE DATA FOR THE FILTER VS. ACTUAL RMS ERROR PLOTS
FILST-CALCULATES THE ERROR STATISTICS
FILTER-DEFINES THE FILTER STATE TRANSITION AND QFD MATRICES
FOUR-IMPLEMENTS THE FFT
INITF-GIVES THE INITIAL FILTER VALUES FOR TDF AND TAF
INPUT3-CREATES THE MEASUREMENT ARRAY FOR INCORPORATION INTO THE
KALMAN FILTER
PLTA-PLOTS THE FILTER VS. ACTUAL ERROR DATA
PLTH-PLOTS THE MEAN ERRORS +/- 1 STANDARD DEVIATION
PROP-PROPAGATES THE TRUTH MODEL STATES
PROPF-PROPAGATES THE FILTER STATES
QDEST-IMPLEMENTS THE QFD ESTIMATION ROUTINE
SHIFT-IMPLEMENTS A SPATIAL PHASE SHIFT IN THE FREQUENCY DOMAIN
SMOOTH-PERFORMS THE EXPONENTIAL SMOOTHING OF THE INTENSITY PATTERN
SPTH-SETS UP THE SPATIAL NOISE CORRELATION COEFFICIENT MATRIX
STATFM-CALCULATES THE ERRORS AT T(I-) FOR USE BY FILST
STATFP-CALCULATES THE ERRORS AT T(I+) FOR USE BY FILST
TRUTH-DEFINES THE TRUTH MODEL STATE TRANSITION AND QD MATRIX
UPCKKF-IMPLEMENTS THE KALMAN FILTER MEASUREMENT UPDATE EQUATION

DATA STRUCTURES TO GATHER STATISTICS ON FILTER
TRACKED CAPABILITY

XFME IS THE ERROR BETWEEN THE PREDICTED X DYNAMIC LOCATION
AT A PARTICULAR TIME AND THE TRUTH MODEL TRUE

```

C      X DYNAMIC LOCATION
CF
C      XFME2 IS THE SQUARE OF THE XFME
C
C      NOTE THAT XFME AND XFME2 ARE ARRAYS WHICH ARE DIMENSIONED TO
C      REAL XFME(4,200),XFME2(4,200),CNME(4,200)
C      REAL CNME2(4,200),XFPMS(4,200),XFMPS(4,200)
C      REAL CNMMS(4,200),CMNPS(4,200)
C          BE 2X20 THE FIRST ROW IN EACH IS USED FOR THE X
C          DIRECTION WHILE THE SECOND ROW IS FOR THE Y DIRECTION
C
C      CNME IS THE ERROR IN THE PREDICTED LOCATION OF THE CENTROID AT
C          A PARTICULAR MINUS TIME COMPARED TO THE TRUTH MODEL
C      CNMF2 IS THE SQUARE OF CNME
C
C      NOTE AGAIN THE DIMENSION OF CNME AND CNME2C
C
C      XFPE IS THE ERROR BETWEEN THE UPDATED DYNAMIC LOCATION AT A
C          PARTICULAR PLUS TIME AND THE TRUTH MODEL TRUE DYNAMIC
C      XFPE2 IS THE SQUARE OF XFPE
C
C      NOTE THE DIMENSIONALITY OF XFPE,XFPE2 FOR THE SAME REASONS AS
C      REAL XFPE(4,200),XFPE2(4,200),CNPE(4,200)
C      REAL CNPE2(4,200)
C      REAL XFPMS(4,200),XFPMS(4,200),CNPMS(4,200),CNPPS(4,200)
C      REAL PFPST(8,200),PFMST(8,200)
C
C      CNPF IS THE CENTROID ERROR AT THE PLUS TIME
C
C      FILTERS DATA STRUCTURES
C
C      PHIF IS THE STATE TRANSITION MATRIX FOR THE KALMAN FILTER
C          -SEE SUBROUTINE FILTER
C      QFD IS THE RESULT OF THE INTEGRAL TERM IN THE PROPAGATION
C          -OF THE COV MATRIX SEE SUBROUTINE PROPF
C      PFP IS THE FILTERS COVARIANCE MATRIX PLUS- AFTER INCORPORATION
C          -OF A MEASUREMENT
C      PFM IS THE FILTERS COVARIANCE MATRIX MINUS AFTER PROPAGATION
C          -BUT PRIOR TO MEASUREMENT INCORPORATION
C      XFP IS THE FILTER STATE VECTOR PLUS
C      XFM IS THE FILTER STATE VECTOR MINUS
C
C      REAL PHIF(8,8),QFD(8,8),PFP(8,8),PFM(8,8),XFP(8),XFM(8),TIME
C
C      Z IS THE KALMAN FILTER MEASUREMENT VECTOR
C
C      REAL Z(64)
C
C      DIMENSION NAME(45)
C
C      ARRAY NAME STORES THE PLOT TITLES
C
C      DATA NAME(1)/40FILTER V ACTUAL ERROR (X-POSITION) /
C      DATA NAME(5)/40FILTER VS ACTUAL ERROR (Y-POSITION) /
C      DATA NAME(9)/40ESTIMATED X-MINUS POSITION (+/-) SIGMA /
C      DATA NAME(13)/40ESTIMATED Y-MINUS POSITION (+/-) SIGMA /

```



```

DATA NAME(17)/40HESTIMATED Y-PLUS POSITION (+/-) SIGMA /
DATA NAME(21)/40HESTIMATED Y-PLUS POSITION (+/-) SIGMA /
DATA NAME(25)/50HESTIMATED X-MINUS CENTROID POSITION (+/-) SIGMA
/
DATA NAME(30)/50HESTIMATED Y-MINUS CENTROID POSITION (+/-) SIGMA
/
DATA NAME(35)/50HESTIMATED X-PLUS CENTROID POSITION (+/-) SIGMA
/
DATA NAME(40)/50HESTIMATED Y-PLUS CENTROID POSITION (+/-) SIGMA
/

```

C
C
C

DATA NR/24,24/

C
C
C
C

INITIALIZE THE FILTER DATA STRUCTURES

```

XVELT0=-1000.
YVELT0=0.
ZVELT0=0.
DT=(1./30.)
ASPR0=1.
SIGM0=SQRT(2.)

```

C
C

FOR A ONE HOT SPOT SIMULATION SET NUMHS=1
NUMHS=3

WRITE(6,576)NUMHS

576 FORMAT(1X,* NUMBER OF HOT SPOTS=*,I4)

C
C

IF USING THE EXTERNAL TRAJECTORY SET TRAGEN=2
RAGEN=1

IF (TRAGEN.EQ.2) WRITE(6,551)

551 FORMAT(1X,* EXTERNAL TRAJECTORY*)

C
C

SELECT THE POINT IN THE EXTERNAL TAPE TO START THE RUN

C
C

IF (TRAGEN.NE.2) GO TO 553

STRUN IS THE PARAMETER WHICH SETS THE START POINT
STRUN=1

WRITE(6,552)STRUN

552 FORMAT(1X,* RUN STARTED AT *,I4)

553 CONTINUE

C
C

THE FOLLOWING IS A LIST OF THE CORRELATION METHODS USED

CORPL=1 FFT METHOD

CORPL=2 PHASE CORRELATION METHOD

CORPL=3 DIRECT METHOD 2-LEVEL QUANTIZATION

CORPL=4 DIRECT METHOD 6-LEVEL QUANTIZATION

C
C

CORPL=1

C
C

WRITE(6,564)CORPL

564 FORMAT(1X,*CORRELATOR=*,I4)

C
C

IF PLOTS ARE DESIRED SET IPLOT = 2

```

      IPILOT=1
C
C      WRITE(6,41)
41      FORMAT(7,1X,* CASE 000*)
C
C
C      SET THE CORRELATOR THRESHOLD
      THRESH=.3
C
C
C      IF THE INTERNAL TRAJECTORY IS BEING USED TRATE SETS THE
C      DESIRED G MANEUVER
C      A TURN RATE(TRATE) = .0196 YIELDS A 2-G TURN
C      A TURN RATE = .078 YIELDS A 10-G TURN
C      A TURN RATE = .196 YIELDS A 20-G TURN
      TRATE=.0196
      WRITE(6,550) TRATE
550      FORMAT(1X,* THE TURN RATE IS *,F9.4//)
C
C      TDF IS THE FILTER ACCELERATION TIME CONSTANT
      DATA TDF/3.5/
      WRITE(6,31) TDF
31      FORMAT(1X,* TDF=*,E14.6)
C
C      THE FILTER MEASUREMENT COVARIANCE MATRIX (RFIL) IS DEFINED
C      ACCORDING TO THE CORRELATION METHOD STATISTICS
      DATA RFIL/.00436,0.,0.,.00579/
C      IF(CORRL.EQ.2) RFIL(1,1)=.472
C      IF(CORRL.EQ.2) RFIL(2,2)=.374
C      IF(CORRL.EQ.3) RFIL(1,1)=.0043
C      IF(CORRL.EQ.3) RFIL(2,2)=.00540
C      IF(CORRL.EQ.4) RFIL(1,1)=.00733
C      IF(CORRL.EQ.4) RFIL(2,2)=.0071
C
C*****
C**              I N I T I A L I Z A T I O N              **
C*****
C
      CALL RANSET(12345)
54321 WRITE(6,3707)
3707  FORMAT(1X,*GAUSSIAN TARGET COVARIANCE VALUE*)
      READ(5,550) COV
550    FORMAT(F6.2)
      IF(COV(5).NE.0) GO TO 5421
      WRITE(6,3701)
3701  FORMAT(1X,*NUMBER OF ZEROS TO PAD*)
      READ(5,561) NZ
561    FORMAT(I2)
      NZM=25-NZ
      WRITE(6,3702)
3702  FORMAT(1X,*NUMBER OF FRAMES*)
      READ(5,562) NFRAMES
562    FORMAT(I3)
      WRITE(6,4023)
4023  FORMAT(1X,*NUMBER OF SIMULATIONS*)
      READ(5,561) NRUNS
      WRITE(6,3703)

```

```

37 3  FORMAT( X, *ALPHA FOR SMOOTHING*)
      READ(5,560) ALPHA
37 4  WRITE(6,37 5)
37 5  FORMAT(1X, *NUMBER OF HIGH FREQ COMPONENTS TO ZERO*)
      READ(5,561) NFREQ
      ISF=14-NFREQ
      IEF=12-NFREQ
      WRITE(6,37 6)
37 6  FORMAT(1X, *INPUT MEASUREMENT ERROR VARIANCE*)
      READ(5,560) VARIN
562  FORMAT(2F6.2)
      READ(5,563) X0
      WRITE(6,37 7) X0
37 7  FORMAT(1X, *INITIAL X POSITION*, E16.7)
563  FORMAT(F10.2)
      READ(5,563) Y0
      WRITE(6,37 8) Y0
37 8  FORMAT(1X, *INITIAL Y POSITION*, E16.7)
      READ(5,563) Z0
      WRITE(6,37 9) Z0
37 9  FORMAT(1X, *INITIAL Z POSITION*, E16.7)

```

```

C
C      CALL INITF(TAF,VARDF,VARAF)
C
C      DEFINE TRUE TARGET AS 3 INDEPENDENT GAUSSIAN FUNCTIONS WITH
C      VARIANCE=COV.
C
      S(1)=1./COV
      S(2)=0.
      S(3)=0.
      S(4)=S(1)
      S(5)=S(1)
      S(6)=0.
      S(7)=0.
      S(8)=S(1)
      S(9)=S(1)
      S(10)=0.
      S(11)=0.
      S(12)=S(1)
C
C      INITIALIZE TARGET INTENSITY ASSUMING
C      3 CIRCULAR CROSS-SECTION GAUSSIAN TARGET.
C
      IMAX(1)=20.
      IMAX(2)=20.
      IMAX(3)=20.
C
C      DEFINE TRUTH MODEL DYNAMICS
C
      WRITE(6,965)
965  FORMAT(2X, *STD. DEV OF TRUTH MODEL ATMOSPHERIC JITTER*)
      READ(5,6666) SIGDT
6666  FORMAT(F5.3)
C

```

```

C IF ICHQ = 1 QFD IS ADAPTIVELY ESTIMATED
  READ(5,561) ICHQ
  IF(ICHQ.NE.1) GO TO 570
  WRITE(6,571)
571  FORMAT(1X,/, ' ADAPTIVELY ESTIMATE QFD',//)
570  CONTINUE
C
  CALL TRUTH(FHIT,QDSOOT,4,SIGDT,DT)
C
C SET THE EXTERNAL MATRICES TO ZERO
C
DO 554 I=1,200
  H1A(I)=0.
  H1B(I)=0.
  H2A(I)=0.
  H2B(I)=0.
  H3A(I)=0.
  H3B(I)=0.
  VMAXEX(I)=0.
  FANEX(I)=0.
  COSW(I)=0.
  TIMEX(I)=0.
  EXUT(1,I)=0.
  EXUT(2,I)=0.
554  CONTINUE
C
C
C
C INITIALIZE THE FILTER ERROR MATRICES TO ZERO
C
DO 21 I=1,4
DO 21 J=1,NFFAMES
  XFME(I,J)=0.
  XFME2(I,J)=0.
  CNME(I,J)=0.
  CNME2(I,J)=0.
  XFPE(I,J)=0.
  XFPE2(I,J)=0.
  CNPE(I,J)=0.
21  CNPE2(I,J)=0.
C
DO 22 I=1,8
DO 22 J=1,NFFAMES
  PFPS(I,J)=0.
  PFMST(I,J)=0.
22  CONTINUE
C USING FIRST AND SECOND NEAREST NEIGHBOR DETERMINE THE CHOLESKY
C SQUARE ROOT, R, OF THE MEASUREMENT COVARIANCE MATRIX, R
C
  CALL SPTR(VARM,8)
C THIS LOOP MAKES SPATIALLY CORRELATED/UNCORRELATED NOISE
C COMMENT THE NEXT FOUR LINES IF WANT SPATIAL CORRELATION
C DO 6428 I=1,64
C DO 6428 J=1,64
C R(I,J)=0.
C IF(I.EQ.J) R(I,J)=VARM
6428 CONTINUE
C MODCOMP VERSION OF CHOLESKY PUTS ROOT BACK INTO CALLING MATRIX

```

```

      CALL CHOLY(5,54)
C
C      READ IN THE INITIAL DATA FROM THE EXTERNAL TRAJECTORY TAPE
C
      IF(ITER.NE.2)GO TO 555
555  CONTINUE
      READ(9,*)ITER,EXUT(1,1),EXUT(2,1),VMAXEX(1),
+COSW(1),RANEX(1),TIMEX(1)
557  FORMAT(I4,6E14.5)
      READ(9,*)HS1A(1),HS1B(1),HS2A(1),HS2B(1),HS3A(1),
+HS3B(1)
559  FORMAT(6E14.5)
      READ(9,*)XPOS,YPOS,ZPOS,XVELTO,YVELTO,ZVELTO,TRATE
556  FORMAT(7E14.5)
C
C      LOOP UNTIL THE START POINT IS REACHED
C
      IF(ITER.NE.STRUN)GO TO 559
      XC=XPOS
      YC=YPOS
      ZC=YPOS
      XDOT=XVELTO
      YDOT=YVELTO
      ZDOT=ZVELTO
      WRITE(6,541)
541  FORMAT(I2,*, INITIAL STARTING CONDITIONS*)
      WRITE(6,542)
542  FORMAT(I4,*,TIME*,T15,*,XPOS*,T23,*,YPOS*,T44,*,ZPOS*,
+T57,*,TURN RATE*,T71,*,HS1A*,T55,*,HS2A*,T39,*,HS3A*,
+T112,*,ROLL ANGLE*)
      WRITE(6,543)TIMEX(1),XC,YC,ZC,TRATE
+HS1A(1),HS2A(1),HS3A(1),COSW(1)
543  FORMAT(I1,F9.3,6E14.5)
555  CONTINUE
C
C
C*****
C**          E N D   I N I T I A L I Z A T I O N          **
C*****
C
C*****
C**          R E G I N   M O N T E C A R L O   S I M U L A T I O N          **
C*****
C
      MAKE NRUNS SIMULATIONS OF NFRAMES EACH FOR MONTE-CARLO ANALYSIS
C
      DO 50 NS=1,NRUNS
C
      WRITE(6,544)NS
544  FORMAT(/,1X,*, SIMULATION NUMBER*,I4)
C
      TIME=0.
      XSHIFT=0.
      YSHIFT=0.
C INITIALIZE SMOOTHED DATA ARRAY
C
      DO 7 I=1,24

```

```

      DO 7 J=1,24
        SDATA(I,J)=CMPLX(0.,0.)
7
C
C      INITIALIZE TRUTH MODEL STATE VECTOR
C
      DO 71 I=1,4
71      XT(I,1)=0.
          XT(1,1)=X0
          XT(2,1)=Y0
          YT(1,1)=X0
          YT(2,1)=Y0
C
C      INITIALIZE THE FILTERS MATRICES DEFINITION
C      CALL FILTER(DF,VAPDF,TAF,VAPAF,DT,PHIF,QFD,NS)
C      ZERO THE FILTER MATRICES FOR THE NEXT RUN
C
      DO 106 I=1,8
      DO 106 J=1,8
        PFM(I,J)=0.0
        PFP(I,J)=0.0
        XFP(I)=0.0
106      XFM(I)=0.0
C
C      SET UP THE FILTER COVARIANCE MATRIX INITIAL CONDITIONS
      PFP(1,1)=10.
      PFP(2,2)=10.
      PFP(3,3)=2000.
      PFP(4,4)=2000.
      PFP(5,5)=100.
      PFP(6,6)=100.
      PFP(7,7)=.2
      PFP(8,8)=.2
      PFM(1,1)=PFP(1,1)
      PFM(2,2)=PFP(2,2)
      PFM(3,3)=PFP(3,3)
      PFM(4,4)=PFP(4,4)
      PFM(5,5)=PFP(5,5)
      PFM(6,6)=PFP(6,6)
      PFM(7,7)=PFP(7,7)
      PFM(8,8)=PFP(8,8)
C
C
      DO 43 I=1,8
      PFPST(I,1)=PFP(I,1)+PFPST(I,1)
43      CONTINUE
C
C
C      INITIAL CONDITIONS ON DYNAMIC STATES
C
      XFP(1)=X0
      XFP(2)=Y0
      XFM(1)=XFP(1)
      XFM(2)=XFP(2)
      RHOR=SQRT(X0**2+Y0**2+20**2)
      RHOR=(X0**2+20**2)
      RANGE=(RHOR+Y0**2)

```

```

      XFP(3)=(-ZG*VELTO+XG*ZVELTO)/(RHOR*.00002)
      RHOR=SQRT(RHOC)
      XFP(4)=(RHOR*YVELTO-YG*((XG*VELTO+ZG*ZVELTO)/RHOR))/(RANGE*
      *.00002)
      RANGE=SQRT(RANGE)
      VMAX=SQRT(XVELTO**2+YVELTO**2+ZVELTO**2)/(RANGE*.00002)
      XFM(3)=XFP(3)
      XFM(4)=XFP(4)
      XFP(5)=55.3
      XFP(6)=-2.0
C
      DO 15 I=1,3
        BD(I,1)=0.
        BD(I,2)=0.
15    CONTINUE
C
C
      BD(1,1)=DT
      BD(2,2)=DT
      UT(1,1)=XFP(3)
      UT(2,1)=XFP(4)
C      DEFINE UPPER-LEFT CORNER OF FOV
C
      X=XFP(1)-4.
      Y=XFP(2)-4.
C
C      ZERO THE VALUES NEEDED TO ESTIMATE QFD
      TRXXT=0.
      TRXXTO=0.
      QPRINT=0
C
C      TRACK TARGET FOR NFRAME FRAMES (TIME SLICES)
C.....
      DO 90 NR=1,NFRAMES
C.....
C
C      DEFINE GAUSSIAN PEAK LOCATIONS BASED ON CENTROID POSITION, YT
C
      YMAX(1)=YT(2,1)
      XMAX(1)=YT(1,1)
      IF(NUMHS.EQ.1)GO TO 72
      YMAX(1)=YT(2,1)+HS1R(NR)
      XMAX(1)=YT(1,1)-HS1A(NR)
      XMAX(2)=YT(1,1)-HS2A(NR)
      XMAX(3)=YT(1,1)-HS3A(NR)
      YMAX(2)=YT(2,1)+HS2R(NR)
      YMAX(3)=YT(2,1)+HS3B(NR)
72    CONTINUE
C
C      GET MEASUREMENT NOISE ARRAY
C
      CALL NOISE(W,64)
      CALL MULT(P,W,64,64,1,V)
C
C      GET MEASUREMENT DATA
C

```

```

C      CALL IDEAL(XMAX,XMAX,YMAX,N,Z,X,Y,DATA,DX,DY,SIGMS,RANGED,RANGE,
C      +UT,VMAX,ASPRC)
C      CALL INPUT3(XMAX,S,XMAX,24,X,Y,DATA,CENX,CENY,YMAX,
C      +SIGMS,RANGED,RANGE,UT,VMAX,ASPRC,NUMHS)
C
C      ADD CORRELATED MEASUREMENT NOISE TO CENTER 3X3 PIXEL DATA
C
C      DO 4 I=1,3
C      DO 4 J=1,3
C      DATA(I+2,J+2)=DATA(I+2,J+2)+CMPLX(V(2*(I-1)+J),0.0)
C      CONTINUE
C
C      ADD UNCORRELATED NOISE TO MEASUREMENT DATA OUTSIDE CENTER
C      3X3 PIXEL AREA.
C
C      CALL NOISE(UC,576)
C      DO 6 I=1,24
C      DO 6 J=1,24
C      IF(I.GE.9.AND.I.LE.16.AND.J.GE.9.AND.J.LE.16) GO TO 6
C      IF((I.LE.N/2).OR.(J.LE.N/2).OR.(I.GE.N/2).OR.(J.GE.N/2)) GO TO 6
C      DATA(I,J)=DATA(I,J)+CMPLX(UC(24*(I-1)+J),0.0)*SQRT(VARM)
C      CONTINUE
C
C      CREATE THE MEASUREMENT VECTOR FOR THE FILTER UPDATE
C
C      K=0
C      DO 101 I=3,16
C      DO 101 J=3,16
C      K=K+1
C      101 Z(K)=REAL(DATA(I,J))
C
C      IF(NE.NE.1) CALL CORREL(NN,DATA,SDATA,XCENT,YCENT,X,Y,THRESH,
C      +CCR,L)
C      Z1(1)=XCENT
C      Z1(2)=YCENT
C
C      GO CALCULATE THE ERRORS OF THE FILTERS ESTIMATE PRIOR TO
C      MEASUREMENT INCORPORATION
C      CALL STATFM(XFME,XFME2,CNME,CNME2,XFM,XT,YT,NR,NFRAMES,
C      +PFM,PFMST,UT)
C
C      INCORPORATE MEASUREMENT
C
C      IF(NR.EQ.1) GO TO 164
C      CALL UPCORF(Z1,XFP,XFM,2FP,PFM,2FIL,UPD)
C
C      IF(NR.NE.1)GO TO 697
C      WRITE(6,696)(XFP(I),I=1,3)
C      676 FORMAT(1X,3XFP,8F14.5)
C      637 CONTINUE
C      CALCULATE THE ERRORS FOR THE FILTER AFTER THE INCORPORATION
C      OF THE MEASUREMENT
C      CALL STATFP(XFPE,XFPE2,CNPE,CNPE2,XT,YT,XFP,NR,NFRAMES,
C      +PEP,PEPST,UT)
C
C      SET THE TIME FOR THE ADAPTATION TO START

```



```

C
  IF(TIME.LT.0.5)GO TO 44
  IF(ICHQ.EQ.0)CALL QDEST(SVPFP,PPF,U'D,TPXXT,TRXXTO
+ ,QFD,TIME,QPRINT,NP)
44  CONTINUE
C
C   COMPUTE THE SHIFT INFORMATION FROM THE CENTER OF FOV
XSHIFT=X-XFP(1)+4.-XFP(7)
YSHIFT=Y-XFP(2)+4.-XFP(8)
C
C   SHIFT THE DATA ARRAY APPROPRIATELY
C
C   GET FORWARD FFT
C
164  CALL FOURT(DATA,NN,2,-1,1,WORK)
C
C   FILTER DESIRED FREQUENCY COMPONENTS OUT
C
  IF(NFREQ.GT.12) NFREQ=12
  IF(NFREQ.LE.0) GO TO 3756
  DO 6 I=1SF,IEF
  DO 6 J=1,24
    DATA(I,J)=CMPLX(0.,0.)
    DATA(J,I)=CMPLX(0.,0.)
R
3756  CONTINUE
C
C   ASSUME IF NP=1 THAT THE DATA IS CENTERED
  IF(NP.NE.1) CALL SHIFT(DATA,2,XSHIFT,YSHIFT)
  CALL SMOOTH(DATA,SDATA,ALPHA,24,NP)
  TIME=TIME + DT
  CALL PROPF(PHIF,QFD,PPF,PFM,XFP,XFM,NS,SVPFP,ICHQ)
  X=XFM(1)-4.
  Y=XFM(2)-4.
C
  PROPAGATE TRUTH MODEL STATE ONE FRAME
  CALL PROCP(PHIT,QDROOT,H,XT,YT,2,2,UT,BD,
+ TIME,DT,TRATE,X0,Y0,Z0,VMAX,RANGE,TRAGEN,NR,NS,EXUT,
+ HS1A,HS1B,HS2A,HS2B,HS3A,HS3B,VMAXEX,RANEX,TIMEX,COSW)
C
  IF(NS.NE.1)GO TO 90
  WRITE(6,694)(XFM(I),I=1,8)
694  FORMAT(/,1X,*,XFM*,8E14.6)
  WRITE(6,695)(XT(I,1),I=1,8)
695  FORMAT(1X,*,XT*,8E14.6)
C
90  CONTINUE
C*****
C**      E N D   M O N T E C A R L O   S I M U L A T I O N      **
C*****
C
C   CALCULATE MEAN AND VARIANCE STATISTICS
C
  CALL FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,NPUNS,
+ NFRAMES,PPFST,PFMST,XFMMS,XFMPS,XFPM3,XFPPS,CNMMS,CNMP3,CNPM3,CNPP
+ 3)
C
  CALL FILPT(PFPST,PFMST,XFPE2,XFME2,NFRAMES,TIMPL,ACTPL,
+ FILPL,FILPY,ACTPY,TIMR)

```

```

C
NIM1=NFRAMES+2
NIM=2*NFRAMES+2
C
IF(IPLT.NE.1)GO TO 565
CALL PLOTS(0.,0.,0)
CALL PLOT(3.,3.,-3)
C
C
C
A 1 IN PLTB YIELDS X STATISTICS A 2 YIELDS Y
CALL PLTA(TIMPL,ACTPL,FILPL,NFRAMES,NIM,1,NAME)
CALL PLTA(TIMPL,ACTPY,FILPY,NFRAMES,NIM,5,NAME)
C
CALL PLTB(TIMR,XFMMS,XFMPS,XFME,NIM1,NFRAMES,1,NAME,0,NIM,
+PTA,PTB,PTC,PTD)
CALL PLTB(TIMR,XFMMS,XFMPS,XFME,NIM1,NFRAMES,2,NAME,13,NIM,
+PTA,PTB,PTC,PTD)
CALL PLTB(TIMR,XFMPS,XFPPS,XFPE,NIM1,NFRAMES,1,NAME,17,NIM,
+PTA,PTB,PTC,PTD)
CALL PLTB(TIMR,XFMPS,XFPPS,XFPE,NIM1,NFRAMES,2,NAME,21,NIM,
+PTA,PTB,PTC,PTD)
CALL PLTB(TIMR,CNMMS,CNMPS,CNME,NIM1,NFRAMES,1,NAME,25,NIM,
+PTA,PTB,PTC,PTD)
CALL PLTB(TIMR,CNMMS,CNMPS,CNME,NIM1,NFRAMES,2,NAME,30,NIM,
+PTA,PTB,PTC,PTD)
CALL PLTB(TIMR,CNPMMS,CNPPS,CNPE,NIM1,NFRAMES,1,NAME,35,NIM,
+PTA,PTB,PTC,PTD)
CALL PLTB(TIMR,CNPMMS,CNPPS,CNPE,NIM1,NFRAMES,2,NAME,40,NIM,
+PTA,PTB,PTC,PTD)
C
C
CALL SYMBOL(0.,0.,0.25,0HCASE P1 ,0.,0)
CALL PLOTE
565 CONTINUE
C
WRITE(6,9987) NRUNS,NFRAMES,NZ,NFREQ,COV,VARM,ALPHA,
SIGDT
9987 FORMAT(1H1,T10,*,RUNS=*,I2,T33,*,FRAMES=*,I4,T73,*,NUMBER ZERO PAD=*,
+I1,T10,*,NUMBER FREQ ZEROED=*,I2,/,T10,*,GAUSSIAN COVARIANCE=*,
+F5.2,T33,*,MEASUREMENT NOISE VARIANCE=*,F5.1,T73,*,SMOOTHING ALPHA
+*,F7.3,/,T10,
+*,TRUTH MODEL UNCERTAINTY=*,F7.3,///)
WRITE(6,45)THRESH
45 FORMAT(1X,*, THE CORRELATOR THRESHOLD IS *,F6.2)
GO TO 54321
6421 STOP
END

```

CRITY DETAILS DIAGNOSIS OF PROBLEM

FILTER V	HOLLERITH CONSTANT .GT. 10	CHARACTERS, EXCESS CHARACTERS INITIAL
FILTER V	HOLLERITH CONSTANT .GT. 10	CHARACTERS, EXCESS CHARACTERS INITIAL
ESTIMATE	HOLLERITH CONSTANT .GT. 10	CHARACTERS, EXCESS CHARACTERS INITIAL
ESTIMATE	HOLLERITH CONSTANT .GT. 10	CHARACTERS, EXCESS CHARACTERS INITIAL
ESTIMATE	HOLLERITH CONSTANT .GT. 10	CHARACTERS, EXCESS CHARACTERS INITIAL

```

SUBROUTINE UPCKKF(Z1,XFP,XFM,PEP,PFM,RFIL,UPD)
REAL Z1(2),XFP(8),XFM(8),PEP(8,8),PFM(8,8),RFIL(2,2)
REAL TEMP1(8,2),TEMP2(2,2),TEMINV(2,2),TEMP3(2,8),TEMP4(8,8)
REAL RESI(2),UPD(8)
REAL HT(8,2),H(2,8),GACOK(3,2)

```

```

C
C THIS SUBROUTINE IMPLEMENTS THE KALMAN FILTER MEASUREMENT
C UPDATE EQUATION.
C  $X(TI+) = X(TI-) + K(TI)(Z(TI) - HX(TI-))$ 
C  $P(TI+) = P(TI-) - K H P(TI-)$ 
C WHERE
C  $K(TI) = P(TI-)HT(HP(TI-)HT+R)^{-1}$ 
C
C DO 5 I=1,8
C DO 5 J=1,2
C   H(J,I)=0.
C
C CONTINUE
C H(1,1)=1.
C H(1,7)=1.
C H(2,2)=1.
C H(2,8)=1.
C DO 6 I=1,8
C DO 6 J=1,2
C   H(I,J)=H(J,I)
C
C CONTINUE
C CALL MULT(PFM,HT,8,8,2,TEMP1)
C CALL MULT(H,TEMP1,2,8,2,TEMP2)
C DO 1 I=1,2
C DO 1 J=1,2
C   TEMP2(I,J)=TEMP2(I,J)+RFIL(I,J)
C CALL INVERT(TEMP2,2,TEMINV)
C CALL MULT(HT,TEMINV,8,2,2,TEMP1)
C CALL MULT(PFM,TEMP1,8,8,2,GACOK)
C CALL MULT(H,PFM,2,8,8,TEMP3)
C CALL MULT(H,XFM,2,8,1,RESI)
C CALL MULT(GACOK,TEMP3,3,2,3,TEMP4)
C DO 2 I=1,8
C DO 2 J=1,3
C   PEP(I,J)=PFM(I,J)-TEMP4(I,J)
C
C CONTINUE
C DO 3 I=1,2
C RESI(I)=Z1(I)-PEP(I)
C CALL MULT(GACOK,RESI,3,2,1,UPD)
C DO 4 I=1,8
C XFP(I)=XFM(I)+UPD(I)
C
C
C
C RETURN
C END

```

```

      SUBROUTINE QDEST(SVPFP,PEP,JPD,TRXXT,TRXXT0,QFON,
      +TIME,QPRINT,AS)

```

C

```

      REAL SVPFP(8,8),PEP(8,8),UPD(8,8),DXDXT(8,8)
      REAL QDMAX(8)
      REAL QFON(8,8),QFDO(8,8),SAVE(3,1)
      REAL NTRXXT,TRXXT0,QFACTOR
      INTEGER NFRAMES,QPRINT

```

C

C THIS ROUTINE ESTIMATES QFD USING:

C $QD = (X(TI+) - X(TI-)) * (X(TI+) - X(TI-))T + P(TI+) - PHIF * P(TI-1+) + PHIFT$

C

C THE FIRST TERM IS APPROXIMATED BY

C $(X(TI+) - X(TI-)) * (X(TI+) - X(TI-))T = K(TI)R(TI)R(TI)TK(TI)T$

C

C WHERE:

C $K(TI)$ = THE KALMAN FILTER GAINC $R(TI)$ = THE FILTER RESIDUAL

C

C SVPFP = THE LAST TERM IN THE QD EQUATION

C

C A FADING MEMORY TECHNIQUE IS USED INSTEAD OF TIME AVERAGING

C

C FROM THE FILTER UPDATE $UPD = K(TI) * R(TI)$

C

C USING THIS $\Delta X * \Delta X(T)$ IS OBTAINED

C

DO 80 I=1,8

DO 80 J=1,8

QFDO(I,J)=0.

80

CONTINUE

QDMAX(1)=2.

QDMAX(2)=2.

QDMAX(3)=14.

QDMAX(4)=14.

QDMAX(5)=20.

QDMAX(6)=20.

QDMAX(7)=.5

QDMAX(8)=.5

C

C

NTRXXT=0.

DO 10 I=1,8

DO 10 J=1,8

DXDXT(I,J)=UPD(I)*UPD(J)

10

CONTINUE

C

DO 20 I=1,8

NTRXXT = NTRXXT + DXDXT(I,I)

20

CONTINUE

C

TRXXT0 = TRXXT

TRXXT = .9 * TRXXT0 + .2 * NTRXXT

C

C STORE QFD FROM THE PREVIOUS ESTIMATION

C

DO 30 I=1,8

DO 30 J=1,8

QFDO(I,J)=QFON(I,J)

30

CONTINUE

```

C
C FORM THE NEW ESTIMATE OF QFD
C
      DO 40 I=1,8
      DO 40 J=1,8
        SAVE(I,J)= PF2(I,J) + DXDXT(I,J)
        QFDN(I,J)= SAVE(I,J) - SVPEP(I,J)
40    CONTINUE
C
C INSURE THAT THE NEW QFD IS BOUNDED
C
      DO 50 I=1,8
        QFACTOR=1.
        IF(QFDN(I,I).GT.0.)GO TO 51
        QFACTOR=0.
        QFDN(I,I)=.1
        GO TO 52
51    CONTINUE
        IF(SQRT(QFDN(I,I)).GT.QDMAX(I)) QFACTOR=QDMAX(I)/SQRT(
+QFDN(I,I))
        IF(QFACTOR.GE.1.)GO TO 50
        QFDN(I,I)=QFACTOR**2*QFDN(I,I)
52    CONTINUE
      DO 54 J=1,8
        IF(I.EQ.J) GO TO 54
        QFDN(I,J)=QFDN(I,J)*QFACTOR
54    CONTINUE
C
      DO 55 K=1,8
        IF(I.EQ.K)GO TO 55
        QFDN(K,I)= QFDN(K,I)*QFACTOR
55    CONTINUE
50    CONTINUE
C
C COMBINE THE OLD AND NEW INFORMATION
C
      DO 56 I=1,8
      DO 56 J=1,8
        QFDN(I,J)=.2*QFDN(I,J)+.3*QFDN(I,J)
56    CONTINUE
        IF(NS.NE.1)GO TO 70
        IF(QPRINT.NE.5)GO TO 70
C
        QPRINT=0
        WRITE(6,61) TIME
61    FORMAT(1X,/, * AT TIME=*,F10.4)
        WRITE(6,62)((QFDN(I,J),J=1,3),I=1,3)
62    FORMAT(1X,/, * ESTIMATED QFD IS*,/, (1X,8E14.5))
        WRITE(6,63)TRXXT
63    FORMAT(1X,/, * TRXXT = *,E14.5)
C
70    CONTINUE
        QPRINT=QPRIN +1
        RETURN
      END

```

```

SUBROUTINE STATEM(XFME,XFME2,CNME,CNME2,XFM,XT,YT,NR,NFRAMES,
+PFM,PFMST,UT)
  INTEGER NR,NFRAME
  REAL XFME(4,NFRAMES),XFME2(4,NFRAMES),CNME(4,NFRAMES)
  REAL CNME2(4,NFRAMES)
  REAL XFM(8),XT(3,1),YT(2,1)
  REAL PFMST(8,200),PFM(8,5)
  REAL UT(2,1)

```

C
C
C
C
C
C
C

THIS ROUTINE GATHERS THE INFORMATION THAT WILL BE
REQUIRED TO COMPUTE THE STATISTICS OF THE PREDICTIONS
OF THE FILTER PRIOR TO MEASUREMENT INCORPORATION

FIRST COLLECT THE ERROR IN THE PREDICTED DYNAMIC LOCATION

```

XFME(1,NR)=XFME(1,NR)+XFM(1)-XT(1,1)
XFME(2,NR)=XFME(2,NR)+XFM(2)-XT(2,1)
XFME(3,NR)=XFME(3,NR)+XFM(3)-UT(1,1)
XFME(4,NR)=XFME(4,NR)+XFM(4)-UT(2,1)

```

C
C
C

NOW COLLECT THE SQUARE OF THAT ERROR

```

XFME2(1,NR)=XFME2(1,NR)+(XFM(1)-XT(1,1))**2
XFME2(2,NR)=XFME2(2,NR)+(XFM(2)-XT(2,1))**2
XFME2(3,NR)=XFME2(3,NR)+(XFM(3)-UT(1,1))**2
XFME2(4,NR)=XFME2(4,NR)+(XFM(4)-UT(2,1))**2

```

C
C
C

COLLECT ERROR IN CENTROID PREDICTED LOCATION MINUS

```

CNME(1,NR)=CNME(1,NR)+(XFM(1)+XFM(7)-YT(1,1))
CNME(2,NR)=CNME(2,NR)+(XFM(2)+XFM(8)-YT(2,1))
CNME(3,NR)=CNME(3,NR)+XFM(3)-XT(3,1)
CNME(4,NR)=CNME(4,NR)+XFM(4)-XT(4,1)

```

C
C
C

COLLECT THE SQUARE OF THE ERROR

```

CNME2(1,NR)=CNME2(1,NR)+(XFM(1)+XFM(7)-YT(1,1))**2
CNME2(2,NR)=CNME2(2,NR)+(XFM(2)+XFM(8)-YT(2,1))**2
DO 10 I=1,8

```

10

```

  PFMST(I,NR)= PFMST(I,NR) + PFM(I,I)
CONTINUE
RETURN
END

```

```

SUBROUTINE STATEP(XFPE,XFPE2,CNPE,CNPE2,XT,YT,XFP,NR,NFRAMES,
+PEP,PEPST,UT)

```

```

  INTEGER NR,NFRAMES

```

```

  REAL XFPE(4,NFRAMES),XFPE2(4,NFRAMES),CNPE(4,NFRAMES)

```

```

  REAL CNPE2(4,NFRAMES)

```

```

  REAL UT(2,1)

```

```

  REAL XT(8,1),YT(2,1),XFP(4)

```

```

  REAL PEPST(3,200),PEP(9,9)

```

```

C
C
C
C
C
C
C

```

```

  THIS ROUTINE GATHERS THE INFORMATION THAT WILL BE
  REQUIRED TO COMPUTE THE STATISTICS ON THE FILTERS
  UPDATED STATE ESTIMATES

```

```

  COMPUTE DIFFERENCES THAT WILL BE NEEDED

```

```

  DIF1=XFP(1)-XT(1,1)

```

```

  DIF2=XFP(2)-XT(2,1)

```

```

  DIF3=XFP(1)+XFP(7)-YT(1,1)

```

```

  DIF4=XFP(2)+XFP(8)-YT(2,1)

```

```

  DIF5=XFP(3)-UT(1,1)

```

```

  DIF6=XFP(4)-UT(2,1)

```

```

C
C
C
C

```

```

  FIRST COLLECT THE ERROR IN THE DYNAMIC LOCATION ESTIMATES

```

```

  XFPE(1,NR)=XFPE(1,NR)+DIF1

```

```

  XFPE(2,NR)=XFPE(2,NR)+DIF2

```

```

  XFPE(3,NR)=XFPE(3,NR)+DIF5

```

```

  XFPE(4,NR)=XFPE(4,NR)+DIF6

```

```

C
C
C

```

```

  NOW COLLECT THE SQUARE OF THAT ERROR

```

```

  XFPE2(1,NR)=XFPE2(1,NR)+DIF1**2

```

```

  XFPE2(2,NR)=XFPE2(2,NR)+DIF2**2

```

```

  XFPE2(3,NR)=XFPE2(3,NR)+DIF5**2

```

```

  XFPE2(4,NR)=XFPE2(4,NR)+DIF6**2

```

```

C
C
C

```

```

  NOW COLLECT THE ERROR IN THE CENTROID UPDATE

```

```

  CNPE(1,NR)=CNPE(1,NR)+DIF3

```

```

  CNPE(2,NR)=CNPE(2,NR)+DIF4

```

```

C
C
C

```

```

  NOW COLLECT THE ERROR SQUARED

```

```

  CNPE2(1,NR)=CNPE2(1,NR)+DIF3**2

```

```

  CNPE2(2,NR)=CNPE2(2,NR)+DIF4**2

```

```

  DO 10 I=1,8

```

```

    PEPST(I,NR)=PEPST(I,NR)+PEP(I,I)

```

```

10

```

```

  CONTINUE

```

```

  RETURN

```

```

  END

```

```

SUBROUTINE FILST(XFME,XFME2,CNME,CNME2,XFPE,XFPE2,CNPE,CNPE2,
#NRUNS,NFRAMES,PEPST,PFMST,XFMMS,XFMPS,XFPMS,XFPPS,CNMMS,CNMP,
+CNPMs,CNPP)

```

```

REAL TIME(200)

```

```

REAL TIMEM(200)

```

```

INTEGER NRUNS,NFRAMES

```

```

REAL XFME(4,NFRAMES),XFME2(4,NFRAMES),CNME(4,NFRAMES)

```

```

REAL CNME2(4,NFRAMES)

```

```

REAL XFPE(4,NFRAMES),XFPE2(4,NFRAMES),CNPE(4,NFRAMES)

```

```

REAL CNPE2(4,NFRAMES)

```

```

REAL XFMMS(4,200),XFMPS(4,200),XFPMS(4,200)

```

```

REAL XFPPS(4,200)

```

```

REAL CNMPS(4,200),CNMMS(4,200),CNPMs(4,200)

```

```

REAL CNPPS(4,200)

```

```

REAL PEPST(4,NFRAMES),PFMST(3,NFRAMES)

```

C
C
C
C
C
C
C
C
C
C
C

EXPLANATION OF DATA STRUCTURES

XFME IS THE XPOS OF THE FILTER AT MINUS TIME ERROR

THESE ARE ALL COMPATIBLE WITH THE ABOVE ROUTINES

XFMMS IS THE XPOS MEAN ERROR AT MINUS TIME MINUS SIGMA

XFMPs IS THE XPOS MEAN ERROR AT PLUS TIME PLUS SIGMA

ALL NAMES FOLLOW THIS CODE

THIS ROUTINE COMPUTES THE STATISTICS ON THE FILTER ERRORS

```

DO 1 I=1,4

```

```

DO 1 J=1,NFRAMES

```

```

XFME(I,J)=XFME(I,J)/FLOAT(NRUNS)

```

```

XFPE(I,J)=XFPE(I,J)/FLOAT(NRUNS)

```

```

CNME(I,J)=CNME(I,J)/FLOAT(NRUNS)

```

```

CNPE(I,J)=CNPE(I,J)/FLOAT(NRUNS)

```

```

DIV=FLOAT(NRUNS-1)

```

```

XFME2(I,J)=SQRT((ABS(XFME(I,J)-FLOAT(NRUNS)*(XFME(I,J)**2)))/DIV)

```

```

XFPE2(I,J)=SQRT((ABS(XFPE(I,J)-FLOAT(NRUNS)*(XFPE(I,J)**2)))/DIV)

```

```

CNME2(I,J)=SQRT((ABS(CNME(I,J)-FLOAT(NRUNS)*(CNME(I,J)**2)))/DIV)

```

```

1 CNPE2(I,J)=SQRT((ABS(CNPE(I,J)-FLOAT(NRUNS)*(CNPE(I,J)**2)))/DIV)

```

```

DO 1001 I=1,4

```

```

DO 1001 J=1,NFRAMES

```

```

XFMMS(I,J)=XFME(I,J)-XFME2(I,J)

```

```

XFMPS(I,J)=XFME(I,J)+XFME2(I,J)

```

```

XFPMS(I,J)=XFPE(I,J)-XFPE2(I,J)

```

```

XFPPS(I,J)=XFPE(I,J)+XFPE2(I,J)

```

```

CNMMS(I,J)=CNME(I,J)-CNME2(I,J)

```

```

CNMPS(I,J)=CNME(I,J)+CNME2(I,J)

```

```

CNPMs(I,J)=CNPE(I,J)-CNPE2(I,J)

```

```

1001 CNPPS(I,J)=CNPE(I,J)+CNPE2(I,J)

```

```

DO 12 I=1,8

```

```

DO 12 J=1,NFRAMES

```

```

PEPST(I,J)=SQRT(PEPST(I,J)/FLOAT(NRUNS))

```

```

PFMST(I,J)=SQRT(PFMST(I,J)/FLOAT(NRUNS))

```

```

12 CONTINUE

```

```

WRITE(6,2)

```

```

2 FORMAT(T2,4FAME,T10,4XERR(-),T22,4SXERR(-),T34,4YERR(-),

```

```

#T46,4SYERR(-),T58,4XVELERR(-),T70,4SXVELERR(-),T82

```

```

#,4YVELERR(-),T94,4SYVELERR(-))

```

C


```

      TIMXX=0.
      DO 3 I=1,NFRAMES
        WRITE(6,4) TIMXX,XFME(1,I),XFME2(1,I),XFME(2,I
4      ),XFME2(2,I),XFME(3,I),XFME2(3,I),XFME(4,I),XFME2(4,I)
        FORMAT(T1,F9.3,E12.5)
        TIMXX=TIMXX+(1./30.)
3      CONTINUE
      TIMYY=0.
      WRITE(6,5)
5      FORMAT(T2,*FRAME*,T10,*XERR(+)*,T22,*SXERR(+)*,T34,*YERR(+)*,
      #T46,*SYERR(+)*,T58,*XVELERR(+)*,T70,*SXVELERR(+)*,
      #T82,*YVELERR(+)*,T94,*SYVELERR(+)*
      DO 6 I=1,NFRAMES
        WRITE(6,7) TIMYY,XFPE(1,I),XFPE2(1,I),XFPE(2,I),XFPE2(2,I)
7      #,XFPE(3,I),XFPE2(3,I),XFPE(4,I),XFPE2(4,I)
        FORMAT(T1,F9.3,E12.5)
        TIMYY=TIMYY+(1./30.)
6      CONTINUE
      TIMXX=0.
      WRITE(6,101)
101     FORMAT(T2,*FRAME*,T12,*CNER(-)*,T30,*SCNER(-)*,T55,*YCER(-)*,
      #T68,*SYCER(-)*
      DO 102 I=1,NFRAMES
        WRITE(6,103) TIMXX,CNME(1,I),CNME2(1,I),CNME(2,I),CNME2(2,I)
103     FORMAT(T1,F9.3,T10,E12.5,T23,E12.5,T52,E12.5,T66,E12.5)
        TIMXX=TIMXX+(1./30.)
102     CONTINUE
      TIMXX=0.
      WRITE(6,10)
10      FORMAT(T2,*FRAME*,T12,*CNER(+)*,T30,*SCNER(+)*,T55,*YCER(+)*,
      #T68,*SYCER(+)*
      DO 8 I=1,NFRAMES
        WRITE(6,9) TIMXX,CNPE(1,I),CNPE2(1,I),CNPE(2,I),CNPE2(2,I)
9      FORMAT(T1,F9.3,T10,E12.5,T23,E12.5,T52,E12.5,T66,E12.5)
        TIMXX=TIMXX+(1./30.)
8      CONTINUE
      TIMXX=0.
      WRITE(6,17)
17      FORMAT(/,T2,*FRAME*,T14,*SQRT PFP(1,1)*,T30,*SQRT PFP(2,2)*,
      #T53,*SQRT PFP(3,3)*,T66,*SQRT PFP(4,4)*
      DO 13 J=1,NFRAME
        WRITE(6,14) TIMXX,PFPST(1,J),PFPST(2,J),PFPST(3,J),PFPST(4,J)
14      FORMAT(T1,F9.3,T12,E12.5,T23,E12.5,T52,E12.5,T66,E12.5)
        TIMXX=TIMXX+(1./30.)
13      CONTINUE
      WRITE(6,18)
18      FORMAT(/,T2,*FRAME*,T14,*SQRT PFM(1,1)*,T30,*SQRT PFM(2,2)*,
      #T53,*SQRT PFM(3,3)*,T66,*SQRT PFM(4,4)*
      TIMXX=0.
      DO 15 I=1,NFRAMES
        WRITE(6,16) TIMXX,PFMST(1,I),PFMST(2,I),PFMST(3,I),PFMST(4,I)
16      FORMAT(T1,F9.3,T10,E12.5,T23,E12.5,T52,E12.5,T66,E12.5)
        TIMXX=TIMXX+(1./30.)
15      CONTINUE
C
      ISTART=15
      IFIN=60

```

```

      DO 10 K=1,2
      TIMXX=FLOAT(ISTART)*(1./30)
      TIMMM=FLOAT(IFIN)*(1./30.)
      WRITE(6,14)TIMXX,TIMMM
84    FORMAT(/,1X,* FROM TIME*,F3.3,* TO TIME*,F3.3)
      WRITE(6,15)
85    FORMAT(/,14,* X/Y EPRM*,T13,* X/Y ERPP*,T32,* X/Y CERPM*,
      *T46,* X/Y CERP*,T60,* X/Y SERRM*,T74,* X/Y SERRP*,
      *T88,* X/Y SCERPM*,T102,* X/Y SCERP*)
      C
      C      THE FOLLOWING CALCULATIONS ARE USED TO FIND THE
      C      AVERAGE ERRORS OVER A GIVEN PERIOD
      C      SET ISTART AND IFIN FOR THE DESIRED RANGE
      C
      DO 81 I=1,2
      EPRM=0.
      ERPP=0.
      CERPM=0.
      CERP=0.
      SERRM=0.
      SERRP=0.
      SCERPM=0.
      SCERP=0.
      C
      DO 82 J=ISTART,IFIN
      EPRM=EPRM+XEME(I,J)
      ERPP=ERPP+XEPE(I,J)
      CERPM=CERPM+CXME(I,J)
      CERP=CERP+CNPE(I,J)
      SERRM=SERRM+XEME2(I,J)
      SERRP=SERRP+XEPE2(I,J)
      SCERPM=SCERPM+CXME2(I,J)+SCERPM
      SCERP=SCERP+CNPE2(I,J)
82    CONTINUE
      C
      EPRM=EPRM/FLOAT(IFIN-ISTART+1)
      ERPP=ERPP/FLOAT(IFIN-ISTART+1)
      CERPM=CERPM/FLOAT(IFIN-ISTART+1)
      CERP=CERP/FLOAT(IFIN-ISTART+1)
      SERRM=SERRM/FLOAT(IFIN-ISTART+1)
      SERRP=SERRP/FLOAT(IFIN-ISTART+1)
      SCERPM=SCERPM/FLOAT(IFIN-ISTART+1)
      SCERP=SCERP/FLOAT(IFIN-ISTART+1)
      C
      WRITE(6,13)EPRM,ERPP,CERPM,CERP,SERRM,SERRP,SCERPM,SCERP
83    FORMAT(1X,(5E14.5))
81    CONTINUE
      IF(NFRAMES.GT.145)ISTART=105
      IF(NFRAMES.GT.149)IFIN=150
80    CONTINUE
837   CONTINUE
      C
      C
      C
      C
      C

```

```

DO 110 I=1,NFRAMES
C   TIME(I) IS THE TIME MINUS ARRAY SET FOR MULTIPLE PLOTS
C   TIME(I)=FLOAT(I)-.2
C   IF(I.EQ.1) TIME(I)=1.
C   TIME(I)=FLOAT(I)
110 IF(I.EQ.1) TIME(I)=1.001
C
C   NOTE THE ORDER OF OUTPUT TO TAPE
C
C   MEAN ERROR AT MINUS X POS FOLLOWED BY PLUS X POS
C   MEAN ERROR AT MINUS X POS MINUS SIGMA FOLLOWED BY X POS
C   MEAN ERROR AT MINUS X POS PLUS SIGMA FOLLOWED BY PLUS X POS
C   MEAN ERROR AT MINUS Y POS FOLLOWED BY PLUS Y POS
C   MEAN ERROR AT MINUS Y POS MINUS SIGMA FOLLOWED BY PLUS Y POS
C   MEAN ERROR AT MINUS Y POS PLUS SIGMA FOLLOWED BY PLUS Y POS
C   MEAN ERROR AT MINUS X CEN POS FOLLOWED BY PLUS XCEN
C   MEAN ERROR AT MINUS X CEN POS MINUS SIGMA FOLLOWED BY PLUS X
C   MEAN ERROR AT MINUS X CEN POS PLUS SIGMA FOLLOWED BY PLUS X
C   MEAN ERROR AT MINUS Y CEN POS FOLLOWED BY PLUS Y CEN
C   MEAN ERROR AT MINUS Y CEN POS MINUS SIGMA FOLLOWED BY PLUS Y
C   MEAN ERROR AT MINUS Y CEN POS PLUS SIGMA FOLLOWED BY PLUS Y
C
C   WRITTEN WITH A 6F12.5 RECORDS WHICH IMPLIES EACH PLOTTABLE
C   SET OF POINTS CONSIST OF 14 RECORDS WITH THE LAST RECORD
C   ONLY CONTAINING 2 POINTS INSTEAD OF 6
C
C   WRITE(1,1000) ((TIME(I),XFME(1,I),TIME(I),XFPE(1,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),XFMS(1,I),TIME(I),XFPM(1,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),XFMP(1,I),TIME(I),XFPP(1,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),XFME(2,I),TIME(I),XFPE(2,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),XFMS(2,I),TIME(I),XFPM(2,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),XFMP(2,I),TIME(I),XFPP(2,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),CNME(1,I),TIME(I),CNPE(1,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),CNMS(1,I),TIME(I),CNPM(1,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),CNMP(1,I),TIME(I),CNPP(1,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),CNME(2,I),TIME(I),CNPE(2,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),CNMS(2,I),TIME(I),CNPM(2,I)),I=1,20)
C   WRITE(1,1000) ((TIME(I),CNMP(2,I),TIME(I),CNPP(2,I)),I=1,20)
1000 FORMAT(6F12.5)
C   RETURN
C   END

```

VERITY DETAILS DIAGNOSIS OF PROBLEM

THERE IS NO PATH TO THIS STATEMENT.

```

SUBROUTINE FILPT(PFP,T,PFMST,XFME2,XFPE2,NFRAMES,TIMPL,ACTPL,
+FILPL,FILPY,ACTPY,TIMR)
REAL TIMPL(400),FILPL(400),ACTPL(400),FILPY(400),ACTPY(400)
REAL PFPST(2,NFRAMES),PFMST(3,NFRAMES)
REAL XFME2(4,NFRAMES),XFPE2(4,NFRAMES)
REAL TIM(200)

```

```

C
C THIS SUBROUTINE COMBINES PFP AND PFM ,AND XFME2
C AND XFPE2 FOR PLOTTING PURPOSES
C

```

```

KCOUNT=1
KCOUNT1=1
KCOUNT2=3
KCOUNT3=1
KCOUNT4=1
TIM=0.

```

```

C
C
DO 10 I=1,NFRAMES
FILPL(KCOUNT)=(PFMST(1,I))
FILPL(KCOUNT+1)=(PFPST(1,I))
KCOUNT=KCOUNT+2
10 CONTINUE
C
C

```

```

DO 20 I=1,NFRAMES
ACTPL(KCOUNT1)=XFME2(1,I)
ACTPL(KCOUNT1+1)=XFPE2(1,I)
KCOUNT1=KCOUNT1+2
20 CONTINUE
C

```

```

DO 30 I=1,NFRAMES
IF(I.LT.1.5)GO TO 40
TIMPL(KCOUNT2)=TIM
TIMPL(KCOUNT2+1)=TIMPL(KCOUNT2) + .000001
TIMR(I)=TIM
KCOUNT2=KCOUNT2+2
TIM=TIM+(1./30.)
GO TO 41
40 TIMPL(I)=0.
TIMR(I)=0.
TIMPL(2)=.000001
TIM=TIM + 1./30.
41 CONTINUE
30 CONTINUE
C
C

```

```

DO 50 I=1,NFRAMES
FILPY(KCOUNT3)=PFMST(2,I)
FILPY(KCOUNT3+1)=PFPST(2,I)
KCOUNT3=KCOUNT3+2
50 CONTINUE
C

```

```

DO 60 I=1,NFRAMES
ACTPY(KCOUNT4)=XFME2(2,I)
ACTPY(KCOUNT4+1)=XFPE2(2,I)
KCOUNT4=KCOUNT4+2

```

AC CONTINUE

C

C

RETURN

END

SUBROUTINE TRUTH(PHIT,GD,COT,4,SIGDT,D")
 REAL PHIT(4,4),GD(6,6),4,GD=DOT(P,9),H(2,6)
 REAL GD(6,6)

THE STATE SPACE MODEL IS:

$D(XD)/DT = FT * XD + BU + GT * WT$ AND $YD = H * XD$
 WHERE

$FT = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -A & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & -B & 1 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -R & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & -A & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 & -R & 1 \\ 0 & 0 & 0 & 0 & 0 & 0 & 0 & -R \end{bmatrix}$

$GT = \begin{bmatrix} 1 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 \\ 0 & 0 & G1 & 0 \\ 0 & 0 & G2 & 0 \\ 0 & 0 & G3 & 0 \\ 0 & 0 & 0 & G1 \\ 0 & 0 & 0 & G2 \\ 0 & 0 & 0 & G3 \end{bmatrix}$

AND $HT = \begin{bmatrix} 1 & 0 & 1 & 1 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 1 & 1 & 0 \end{bmatrix}$

$XD = \begin{bmatrix} XT \\ YT \\ X1A \\ X2A \\ X3A \\ Y1A \\ Y2A \\ Y3A \end{bmatrix}$

THE SOLUTION OF THE DYNAMIC EQUATIONS IS

$XD(I+1) = PHIT * XD(I) + SQRT(GD) * WT$

WHERE PHIT=STATE TRANSITION MATRIX
 SIGDT= ATMOS NOISE STANDARD DEVIATION
 WT= GAUSSIAN NOISE VECTOR
 GD= COVARIANCE MATRIX

INSTATE="

```

K=.362109544*SIGOT
TD=1.
A=14.14
B=659.5
DO 1 I=1,NSTATE
DO 1 J=1,NSTATE
  PHIT(I,J)=1.
1  QD(I,J)=0.
C  PHIT(1,1)=EXP(-DT/D)
C  PHIT(2,2)=EXP(-DT/D)
  PHIT(1,1)=1.
  PHIT(2,2)=PHIT(1,1)
  PHIT(3,3)=EXP(-A*D)
  PHIT(4,5)=DT*EXP(-B*DT)
  PHIT(5,5)=EXP(-B*DT)
  PHIT(6,6)=EXP(-A*DT)
  PHIT(7,7)=EXP(-B*DT)
  PHIT(7,5)=DT*EXP(-B*DT)
  PHIT(3,5)=EXP(-B*DT)
  FACT=(K**2)*(A**2)*(B**4)
  FACT1=A-B
  FACT2=A+B
  FACT3=2.*B
  G1=FACT/(FACT1**4)
  G2=FACT/(FACT1**3)
  G3=FACT/(FACT1**2)
  P1=1.-EXP(-2.*A*DT)
  P2=1.-EXP(-FACT2*DT)
  P3=1.-EXP(-2.*B*DT)
  P4=DT*EXP(-FACT2*DT)
  P5=DT*EXP(-2.*B*DT)
  QD(1,1)=0.
  QD(2,2)=QD(1,1)
  QD(3,3)=(G1*P1)/(2.*A)
  QD(3,4)=P2*(G2/FACT2**2-G1/FACT2)-P4*G2/FACT2
  QD(3,5)=G2*P2/FACT2
  QD(4,3)=QD(3,4)
  QD(4,4)=P3*(G1/FACT3-2.*G2/FACT3**2+2.*G3/FACT3**3)-
    P5*(-G2/B+G3*DT/FACT3+2.*G3/FACT3**2)
  QD(4,5)=P3*(G3/FACT3**2-G2/FACT3)-P5*G3/FACT3
  QD(5,3)=QD(3,5)
  QD(5,4)=QD(4,5)
  QD(5,5)=P3*G3/FACT3
DO 2 I=3,5
DO 2 J=3,5
  QD(I+3,J+3)=QD(I,J)
2  CONTINUE
DO 64 I=1,6
DO 64 J=1,6
  QDP(I,J)=QD(I+2,J+2)
CALL CHOLV(QDP,6)
DO 65 I=1,6
DO 65 J=1,6
  QD(I+2,J+2)=QDP(I,J)
DO 5 I=1,6
DO 5 J=1,6
  QD=QD(I,J)=QD(I,J)
5

```

```
1007 FORMAT(1X, 'H(1,2)')
D 3 I=1,2
D 3 J=1,3
3 H(I,J)=1.
H(1,1)=1.
H(1,3)=1.
H(1,4)=1.
H(2,2)=1.
H(2,6)=1.
H(2,7)=1.
RETURN
END
```



```

SUBROUTINE PROP(PHIT,QDRCOOT,H,XT,YT,N,M,UT,HD,
+TIME,DT,TRATE,X0,Y0,Z0,VMAX,RANGE,TRAGEN,NP,NS,EXUT,
+HS1A,HS1H,HS2A,HS2B,HS3A,HS3B,VMAXEX,RANEX,TIMEX,COSW)
  REAL PHIT(N,N),QDRCOOT(N,N),XT(N,1),YT(M,1),H(M,N)
  REAL TEMP1(3,1),TEMP2(3,1),TEMP3(3,1)
  REAL UT(2,1),HD(8,2)
  INTEGER TRAGEN,NR,NC
  REAL EXUT(2,200),HS1A(200),HS1B(200),HS2A(200),HS2B(200)
  REAL HS3A(200),HS3B(200),VMAXEX(200),RANEX(200)
  REAL TIMEX(200),COSW(200)

```

THIS ROUTINE IMPLEMENTS THE STATE TRANSITION EQUATION.

$$XT(I+1) = PHIT * XT(I) + QDRCOOT * WD$$

WHERE XT= STATE VECTOR (NX1)
 PHIT= STATE TRANSITION MATRIX (NXN)
 QDRCOOT= STATE UNCERTAINTY COVARIANCE MATRIX (NXN)
 WD= GAUSSIAN DISTRIBUTED NOISE VECTOR (NX1)

AND THE OUTPUT EQUATION

$$YT = H * XT$$

WHERE YT= MEASUREABLE OUTPUT VECTOR (MX1)
 H= STATE TO OUTPUT MATRIX (MXN)

```

NP=NP+1
IF (TRAGEN.EQ.2) GO TO 400

```

```

YDOT= -1000.
YDOT= 0.
ZDOT= 0.
XPOS= X0 + XDOT*TIME
YPOS= Y0
ZPOS= Z0
VTIME=TIME-(DT/2.0)
IF (TIME.LT.2.0) GO TO 310
YDOT= -1000.*COS(TRATE*(VTIME-2.))
YDOT= 1000.*SIN(TRATE*(VTIME-2.))
XPOS= X0-2000.-((1000/TRATE)*SIN(TRATE*(TIME-2.)))
YPOS= Y0 + ((1000/TRATE)*(1.-COS(TRATE*(TIME-2.))))
ZPOS= Z0
310 CONTINUE
RHOR= XPOS**2 + ZPOS**2
RANGE= RHOR + YPOS**2
UT(1,1)= (-ZPOS*XDOT+XPOS*ZDOT)/(RHOR*.00002)
RHOR= SQRT(RHOR)
UT(2,1)= (PHIT*YDOT-YPOS*((XPOS*XDOT+ZPOS*ZDOT)/RHOR))/(RANGE
+.00002)
RANGE= SQRT(RANGE)
VMAX=SQRT(XDOT**2+YDOT**2+ZDOT**2)/(RANGE*.00002)
GO TO 401

```

```

C
400 CONTINUE

```

```

C      READ IN THE DATA FROM THE EXTERNAL TAPE
C      AND STORE THE DATA FOR THE NEXT RUN
C
      IF(NC.NE.1)GO TO 402
      READ(9,*)ITER,EXUT(1,NP),EXUT(2,NP),VMAXEX(NP)
      *COSW(NP),RANGEX(NP),TIMEY(NP)
403    FORMAT(14,6E14.5)
      READ(9,*)HS1A(NP),HS1B(NP),HS2A(NP),HS2B(NP),HS3A(NP),HS3B(NP)
404    FORMAT(6E14.5)
      READ(9,*)XPOS,YPOS,ZFOS,XDOT,YDOT,ZDOT,TFATE
405    FORMAT(7E14.5)
C
      WRITE(6,411)
411    FORMAT(/,T1,* FRAME*,T12,* ADOT*,T26,* HDOT*,T40,* VMAX*,
      * T51,*ROLL ANGLE*,T60,*RANGE*,T82,* EX TIME*)
      WRITE(6,406)ITER,EXUT(1,NP),EXUT(2,NP),VMAXEX(NP),COSW(NP)
      *RANGEX(NP),TIMEY(NP)
406    FORMAT(1X,I4,6E14.5)
C
      WRITE(6,412)
412    FORMAT(T6,* HS1A*,T21,* HS1B*,T35,*HS2A*,T49,*HS2B*,T63,
      * HS3A*,T77,* HS3B*)
      WRITE(6,407)HS1A(NP),HS1B(NP),HS2A(NP),HS2B(NP),HS3A(NP),HS3B(NP)
407    FORMAT(1X,6E14.5)
408    CONTINUE
C
      UT(1,1)=EXUT(1,NP)
      UT(2,1)=EXUT(2,NP)
      VMAX=VMAXEX(NP)
      RANGE=RANGEX(NP)
C
401    CONTINUE
C
C
C
      CALL NOISE(TEMP1,N)
      CALL MULT(QDRDOT,TEMP1,M,N,1,TEMP2)
      CALL MULT(PHIT,XT,M,N,1,TEMP1)
      CALL MULT(BD,UT,5,2,1,TEMP3)
      DO 1 I=1,N
1      XT(I,1)=TEMP1(I,1) + TEMP2(I,1) + TEMP3(I,1)
      CALL MULT(H,XT,M,N,1,YT)
      RETURN
      END

```

SUBROUTINE INITF(TAF,VARDF,VARAF)

C
C
C
C
C
C
C

THIS ROUTINE CONTROLS INPUTING VALUES NEEDED FOR THE KALMAN
- FILTER

AF=.07072

C
C
C
C
C
C
C
C

THIS IS THE CORRELATION TIME FOR THE ATMOSPHERIC MODEL FOR THE
- FILTER

THE VARIANCE OF THE DYNAMICS FOR THE FILTER

C
C
C

READ(5,2)VARDF

WRITE(6,1)VARDF

1

FORMAT(1X,'VARIANCE OF FILTER DYNAMICS',E14.5)

WRITE(6,3)

3

FORMAT(1X,'VARIANCE OF FILTER ATMOSPHERICS')

READ(5,2)VARAF

2

FORMAT(F7.4)

C
C
C
C
C
C

THE VARIANCE OF THE ATMOSPHERIC JITTER FOR THE FILTER

RETURN

END

```

      SUBROUTINE FILTER(TDF,VARDF,TAF,VARAF,DT,PHIF,QFD,NF)
      REAL PHIF(8,8),QFD(8,8)
      INTEGER NF
C
C      THIS SUBROUTINE DEFINES THE FILTER STATE TRANSITION
      DO 1 I=1,8
      DO 1 J=1,8
          PHIF(I,J)=0.
          QFD(I,J)=0.
1      CONTINUE
C
      PHIF(1,1)=1.
      PHIF(1,5)=0.
      PHIF(1,5)=(TDF**2)*((DT/TDF)-1.0)+(EXP(-DT/TDF))
      PHIF(2,2)=1.
      PHIF(2,4)=PHIF(1,3)
      PHIF(2,6)=PHIF(1,5)
      PHIF(3,3)=1.
      PHIF(3,5)=TDF*(1.-EXP(-DT/TDF))
      PHIF(4,4)=1.
      PHIF(4,6)=PHIF(3,5)
      PHIF(5,5)=EXP(-DT/TDF)
      PHIF(6,6)=PHIF(5,5)
      PHIF(7,7)=EXP(-DT/TAF)
      PHIF(8,8)=PHIF(7,7)
C      MATRIX (PHIF) AND THE DISCRETE NOISE COVARIANCE
C      MATRIX (QFD)
      QFD(1,1)=((2*VARDF*TDF*(DT**3))/3.)-(2*VARDF*(TDF**2)*(DT**2)
+)-(4*VARDF*(TDF**3)*DT*EXP(-DT/TDF)+(2*VARDF*(TDF**3)*DT)-VARDF*(
+TDF**4)*EXP(-2.*DT/TDF)+(VARDF*(TDF**4))
      QFD(1,3)=(VARDF*TDF*(DT**2))+(2*VARDF*(TDF**2)*DT*EXP(-DT/TDF))+(V
+ARDF*(TDF**3))-(2*VARDF*(TDF**3)*EXP(-DT/TDF))-(2*VARDF*(TDF**2)*D
+T)+(VARDF*(TDF**3)*EXP(-2.*DT/TDF))
      QFD(1,5)=(-2.*TDF*VARDF*DT*EXP(-DT/TDF))+(VARDF*(TDF**2))-(VARDF*(
+TDF**2)*EXP(-2.*DT/TDF))
      QFD(2,2)=QFD(1,1)
      QFD(2,4)=QFD(1,3)
      QFD(2,6)=QFD(1,5)
      QFD(3,1)=QFD(1,3)
      QFD(3,3)=(2.*VARDF*TDF*DT)-(3.*(-TDF**2)*VARDF)+(4.*(TDF**2)*VARDF*
+EXP(-DT/TDF))-((TDF**2)*VARDF*EXP(-2.*DT/TDF))
      QFD(3,5)=(VARDF*TDF)-(2.*VARDF*TDF*EXP(-DT/TDF))+(VARDF*TDF*EXP(-2
+.*DT/TDF))
      QFD(4,2)=QFD(2,4)
      QFD(4,4)=QFD(3,3)
      QFD(4,6)=QFD(3,5)
      QFD(5,1)=QFD(1,5)
      QFD(5,3)=QFD(3,5)
      QFD(5,5)=VARDF*(1.-EXP(-2.*DT/TDF))
      QFD(6,2)=QFD(2,6)
      QFD(6,4)=QFD(4,6)
      QFD(6,6)=QFD(5,5)
      QFD(7,7)=VARAF*(1.-EXP(-2.*DT/TAF))
      QFD(8,8)=QFD(7,7)
C
C      WRITE THE MATRIX DEFINITIONS TO THE OUTPUT TAPE
      IF(NF.NE.1)GO TO 20

```

```
      WRITE(6,200) ((PHIF(I,J),J=1,3),I=1,3)
200   FORMAT(1X,3PHIF*,/, (1X,3F14.5))
      WRITE(6,201) ((QFD(I,J),J=1,3),I=1,3)
201   FORMAT(1X,3QFD*,/, (1X,3E14.5))
20   CONTINUE
C
      RETURN
END
```

```

SUBROUTINE PHREF(PHIF,QFD,PEP,PFM,XFD,XFM,NS,SVPEP,ICHO)
REAL PHIF(8,8),QFD(8,8),PEP(8,8),PFM(8,8),XFD(8),XFM(8)
REAL PHIFT(8,8),TEMP1(8,8),TEMP2(8,8)
REAL SVPEP(8,8)
INTEGER ICHO

```

THIS ROUTINE IMPLEMENTS THE STATE TRANSITION
-EQUATIONS FOR THE FILTER

$$XF(I+1) = PH1F + XF(I)$$

PM=PHIF*PFP*SHIFT +QFD

WHERE
PHIF=FILTER STATE TRANSITION MATRIX
XF =FILTER STATE VECTOR
PFM =COV FILTER STATES MINUS
PPF =COV FILTER STATES PLUS
QFD=NOISE COVARIANCE MATRIX

PERFORM FILTER STATE PROPAGATION

```
CALL MULT(PHIF,XFP,3,8,1,XFM)
CALL MUL(PHIF,PFP,3,8,2,TEMP1)
DO 1 I=1,8
DO 1 J=1,8
    PHIFT(I,J)=PHIF(J,I)
CONTINUE
CALL MULT(TEMP1,PHIFT,8,8,0,TEMP2)
```

SAVE PHIF*PFP*PHIF FOR QFD ESTIMATION

```
IF(ICHQ.NE.1)GO TO 377
DO 376 I=1,N
DO 376 J=1,N
SVFFP(I,J)=TEMP2(I,J)
```

375 CONTINUE

377 CONTINUE

C WRITE(6,362)

36 FORMAT(* TEMP2*)

```
C      CALL MPINT(TEMP2,N,N,10)
```

C WHITE (6,375)

370 FORMAT(* QF0*)

C CALL MPF INT (QFD,8,8,10)

CF(NF.NE.1) G2 10 374

```
C      WRITE(6,3/1) ((FFP(I,J),J=1,6),I=1,3)
```

```
371  FORMAT(1X,'PFP',/, (1X,E14.5))
```

374 CONTINUE

00 2 1=1,2

```

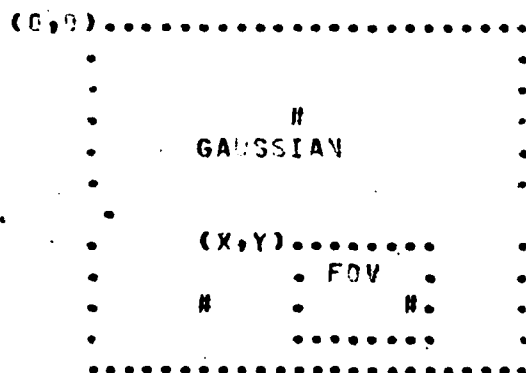
      DO 2 J=1,8
        PFM(I,J)=TEMP2(I,J)+QFD(I,J)
2      CONTINUE
      IF(NS.NE.1) GO TO 375
C      WRITE(6,373) ((PFM(I,J),J=1,8),I=1,9)
373    FORMAT(1X,77,*,PFM*,/, (1X,8E14.5))
375    CONTINUE
      RETURN
      END
  
```

```

SUBROUTINE INPUT3(IMAX,S,XMAX,V,X,Y,DATA,CENX,CENY,YMAX,
+SIGMS,RANGED,RANGE,UT,VMAX,ASPRC,NUMHS)
  REAL IMAX(3),S(12),XMAX(3),YMAX(3)
  COMPLEX DATA(N,4)
  REAL SIGMS,RANGED,RANGE,VMAX,ASPRC
  REAL UT(2,1)
  INTEGER ONEH

```

THIS ROUTINE DETERMINES REAL MODEL INTENSITY AND CENTROID
VALUES FOR AN 8X8 PIXEL FOV. ZERO PADDING IS ACCOMPLISHED BY
CENTERING THE 8X8 PIXEL FOV WITHIN A NULL 8XN SPACE.
THE COORDINATE SYSTEM IS AS DEFINED BELOW:



THE INTENSITY PATTERN IS DEFINED TO BE 3 GAUSSIAN DISTRIBUTION
OF INTENSITY $IMAX(I)$, $I=1,3$ LOCATED AT $XMAX(I)$, $YMAX(I)$, $I=1,3$
WITH COVARIANCE $S(I)$, $I=1,3$. THE UPPER LEFT CORNER OF THE 8X8 PIXEL
FOV IS DEFINED TO BE LOCATION X,Y MICRORAD.

THE INTENSITY AT EACH PIXEL IS DETERMINED BY INTEGRATING THE
INTENSITY OF 25 EQUALLY SPACED SPOTS WITHIN THE PIXEL.

```

SIGPV=SIGMS*(RANGED/RANGE)
PLVEL=SQRT(UT(1,1)**2+UT(2,1)**2)
SNTH=UT(2,1)/PLVEL
CSTH=UT(1,1)/PLVEL
SIGV=(1.+(ASPRC-1.)*PLVEL/VMAX)*SIGPV

```

ZERO OUT FOV SPACE.

```
DO 10 I=1,8
```

```
DO 10 J=1,8
```

```
DATA(I,J)=0.
```

```
SUM=0.
```

```
SUMY=0.
```

```
SUMAVG=0.
```

```
LM=N/2-3
```

```
LP=N/2+4
```

```
DO 1 I=LM,LP
```

```
DO 2 J=LM,LP
```

```
AVG=0.
```

```
DIVIDE PIXEL I,J INTO 25 SEGMENTS
```

```
DO 3 K1=1,5
```



```

      DO 4 K2=1,5
      DELY=(I-9)*1.0+(K1-1)*.2
      DELX=(J-9)*1.0+(K2-1)*.2
      XP=X+DELX
      YP=Y+DELY
      CX1=(XP-XMAX(1))*CSTH
      SX1=(XP-XMAX(1))*SINTH
      CY1=(YP-YMAX(1))*CSTH
      SY1=(YP-YMAX(1))*SINTH
      X1=CX1+SY1
      Y1=CY1-SX1
      ARG1=-.5*((X1/SIGV)**2+(Y1/SIGPV)**2)
      IF (XUMH0.EQ.1)GO TO 70
      CX2=(XP-XMAX(2))*CSTH
      SX2=(XP-XMAX(2))*SINTH
      CY2=(YP-YMAX(2))*CSTH
      SY2=(YP-YMAX(2))*SINTH
      CX3=(XP-XMAX(3))*CSTH
      SX3=(XP-XMAX(3))*SINTH
      CY3=(YP-YMAX(3))*CSTH
      SY3=(YP-YMAX(3))*SINTH
      X2=CX2+SY2
      Y2=CY2-SX2
      X3=CX3+SY3
      Y3=CY3-SX3
      ARG2=-.5*((X2/SIGV)**2+(Y2/SIGPV)**2)
      ARG3=-.5*((X3/SIGV)**2+(Y3/SIGPV)**2)
      FXY=IMAX(1)*EXP(ARG1)+IMAX(2)*EXP(ARG2)+IMAX(3)*EXP(ARG3)
      GO TO 71
70    FXY=IMAX(1)*EXP(ARG1)
71    CONTINUE
      AVG=AVG+FXY
      SUMX=SUMX+XP*FXY
      SUMY=SUMY+YP*FXY
4     CONTINUE
3     CONTINUE
      DATA(I,J)=AVG/25.
C     WRITE(5,100) I,J,DATA(I,J)
10    FORMAT(2X,214,2X,E12.5)
      SUMAVG=SUMAVG+AVG
2     CONTINUE
1     CONTINUE
      CENX=SUMX/SUMAVG
      CENY=SUMY/SUMAVG
      RETURN
      END

```

```

SUBROUTINE CORREL(NN,DATA,TMPLAT,XCENT,YCENT,X,Y,THRESH,CORRL)
COMPLEX TMPLAT(24,24),DATA(24,24)
COMPLEX TEMP(24,24)
COMPLEX WORK(50)
REAL RDATA(24,24)
REAL MAG
INTEGER CORRL
INTEGER NN(2)

C
C THIS SUBROUTINE IMPLEMENTS THE CORRELATION METHODS IN
C FREQUENCY DOMAIN, I.E. THE FFT AND PHASE CORRELATION
C METHOD. IF THE DIRECT METHOD IS BEING USED SUBROUTINE
C
IF(CORRL.GT.2)CALL CORL2(NN,DATA,TMPLAT,XCENT,YCENT,X,Y,CORRL)
IF(CORRL.GT.2)GO TO 20
CALL FOURT(DATA,NN,2,-1,1,WORK)
DO 1 I=1,24
DO 1 J=1,24
TEMP(I,J)=CONJG(TMPLAT(I,J))*DATA(I,J)
IF(CORRL.EQ.2)GO TO 1
XM=REAL(TEMP(I,J))*2
YM=AIMAG(TEMP(I,J))*2
MAG=SQRT(XM+YM)
C
C IN ORDER TO AVOID NUMERICAL DIFFICULTIES THE MAGNITUDE
C IS CHECKED BEFORE THE DIVISION
IF(MAG.LT..000000001)GO TO 101
C
TEMP(I,J)=TEMP(I,J)/MAG
GO TO 1
101 TEMP(I,J)=CMPLX(0.,0.)
1 CONTINUE
CALL FOURT(DATA,NN,2,1,1,WORK)
CALL FOURT(TEMP,NN,2,1,1,WORK)
C
IF(CORRL.EQ.2)GO TO 22
DO 35 I=1,24
DO 35 J=1,24
TEMP(I,J)=TEMP(I,J)/576.
35 DATA(I,J)=DATA(I,J)/576.
22 CONTINUE
C CALL DISPLAY(24,24,DATA)
CALL CHAQUAD(TEMP,24)
C CALL DISPLAY(24,24,DATA)
DO 31 I=1,24
DO 31 J=1,24
RDATA(I,J)=REAL(TEMP(I,J))
31 WRITE(6,112) ((RDATA(I,J),J=1,24),I=1,24)
112 FORMAT(4(1X,6E12.5,/,),/)
CALL CENTRD(X,Y,TEMP,24,XCENT,YCENT,THRESH)
C CALL DISPLAY(24,24,DATA)
20 CONTINUE
XCENT=XCENT-.5
YCENT=YCENT-.5
C WRITE(6,110) XCENT,YCENT
110 FORMAT(1X,*,CENTROID=(*,F7.2,*,*,F7.2,*,*))
C XCUM=XCUM+(XCENT-XSHIFT)
C YCUM=YCUM+(YCENT-YSHIFT)

```

```

C      XCUM2=XCUM2+(XCENT-YSHIFT)**2
C      YCUM2=YCUM2+(YCENT-XSHIFT)**2
1000  CONTINUE
C      CALL STATC(XCUM,YCUM,XCUM2,YCUM2,N1)
      RETURN
      END
    
```

```

SUBROUTINE CORPL2(NN,DATA,TMPLAT,XCENT,YCENT,X,Y,CORPL)
C
COMPLEX TMPLAT(24,24),WORK(57),DATA(24,24),HOLDT(24,24)
REAL COR(24,24),QNT(24,24),QND(24,24),TEMP(24,24),SUMM(17,17)
REAL VA(24,24),MEANT(24,24),VART(24,24)
REAL A(24,24)
INTEGER NN(2)
INTEGER CORPL,PMAX,QMAX
INTEGER P,Q,P3,Q3,N1
REAL Q1,Q2
INTEGER Q5,P2
REAL MEAN'D

C
C THIS SUBROUTINE IMPLEMENTS THE DIRECT CORRELATION METHOD:
C
C
C
C 1=4
HOLD1=0.
HD1=0.
HD2=0.
TOTAL=0.
HOLD2=0.
SUMX=0.
SUMY=0.
SUMA=0.
DO 31 I=1,24
DO 31 J=1,24
A(I,J)=0.
VART(I,J)=0.
VA(I,J)=0.
31 CONTINUE
R=0.
VR=0.
DO 26 I=1,17
DO 26 J=1,17
II=I-1
JJ=J-1
SUMM(II,JJ)=0.
26 CONTINUE
DO 25 I=1,24
DO 25 J=1,24
HOLDT(I,J)=TMPLAT(I,J)
25 CONTINUE
C
CALL FOURT(HOLDT,NN,2,1,1,WORK)
DO 27 I=1,24
DO 27 J=1,24
HOLDT(I,J)=HOLDT(I,J)/575.
27 CONTINUE
C
C THE MEAN FOR THE DATA IS CALCULATED ONE TIME
C THE MEAN FOR THE TEMPLATE IS CALCULATED FOR EACH P,Q VALUE
C THE VARIANCE CALCULATIONS ARE ONLY REQUIRED FOR THE
C THE 6-LEVEL QUANTITIZER
C
DO 30 P3=5,13
DO 30 Q3=5,13

```

```

      Q=Q3-1
      P=P3-1
      DO 1 I=1,8
      DO 1 J=1,8
C
      A(P,Q)=A(P,Q)+(REAL(HOLDT(I+P,J+Q)))
      IF(CORRL.EQ.4)VA(P,Q)=VA(P,Q)+REAL(HOLDT(I+P,J+Q))*2
      IF(N1.NE.4)GO TO 121
      H=B+(REAL(DATA A(I+8,J+8)))
      IF(CORRL.EQ.4)VB=VB+(REAL(DATA A(I+8,J+8)))*2
121  CONTINUE
C
1  CONTINUE
C
      MEANT(P,Q)=A(P,Q)/64.0
      IF(CORRL.EQ.4)VART(P,Q)=SQRT(ABS((VA(P,Q)/64.0)-(MEANT(P,Q)**2)))
      IF(N1.NE.4)GO TO 122
      MEAND=B/64.0
      IF(CORRL.EQ.4)VARP=SQRT(ABS((VB/64.0)-(MEAND**2)))
122  CONTINUE
      (I1=N1+1
30  CONTINUE
C
      IF(CORRL.EQ.4)GO TO 80
C
C  CALCULATE THE QUANTIZED VALUES BASED ON THE MEAN
C  CALCULATIONS FOR THE 2-LEVEL QUANTIZER
C
      DO 15 Q3=5,13
      DO 15 P3=5,13
      Q=Q3-1
      P=P3-1
      DO 16 J=1,8
      DO 16 I=1,8
C
      QNT(I+P,J+Q)=-1.0
      IF(REAL(HOLDT(I+P,J+Q)).GT.MEANT(P,Q))QNT(I+P,J+Q)=1.0
      QND(I+8,J+8)=-1.0
      IF(REAL(DATA A(I+8,J+8)).GT.MEAND)QND(I+8,J+8)=1.0
      SUMM(P,Q)=SUMM(P,Q)+QNT(I+P,J+Q)+QND(I+8,J+8)
16  CONTINUE
C  CHECK FOR THE MAX CORRELATION VALUE
      IF(SUMM(P,Q).LT.TOTAL)GO TO 51
      PMAX=P
      QMAX=Q
      TOTAL=SUMM(P,Q)
      DO 3 I=1,8
      DO 3 J=1,8
      COR(I+PMAX,J+QMAX)=(QNT(I+PMAX,J+QMAX)+QND(I+8,J+8))/64.0
3  CONTINUE
51  CONTINUE
15  CONTINUE
      GO TO 80
50  CONTINUE
C
C  PERFORM THE CALCULATIONS FOR THE 6-LEVEL QUANTIZER
C

```

```

      DO 17 Q5=5,13
      DO 17 P5=5,13
      HD1=0.
      HD2=0.
      P=P5-1
      Q=Q5-1
      D 18 J=1,8
      D 18 I=1,8
      Q1=(REAL(HOLD1(P+1,Q+J))-MEAND)/VARP(P,Q)
      IF(Q1.GT.0.6)GO TO 81
      QNT(I+P,J+Q)=-1.0
C
      IF(Q1.LT.-1.0)QNT(I+P,J+Q)=-3.0
      IF(Q1.LT.-1.4)QNT(I+P,J+Q)=-5.0
      GO TO 82
81  QNT(I+P,J+Q)=1.0
      IF(Q1.GT.0.6)QNT(I+P,J+Q)=3.0
      IF(Q1.GT.1.4)QNT(I+P,J+Q)=6.0
82  CONTINUE
C
      Q2=(REAL(DATA(I+8,J+8))-MEAND)/VARQ
      IF(Q2.GT.0.6)GO TO 83
      QND(I+8,J+8)=-1.0
      IF(Q2.LT.-0.6)QND(I+8,J+8)=-3.0
      IF(Q2.LT.-1.4)QND(I+8,J+8)=-5.0
      GO TO 84
83  QND(I+8,J+8)=1.0
      IF(Q2.GT.0.6)QND(I+8,J+8)=3.0
      IF(Q2.GT.1.4)QND(I+8,J+8)=6.0
84  CONTINUE
      HD1=HD1+QNT(I+P,Q+J)**2
      HD2=HD2+QND(I+8,J+8)**2
C
C
10  CONTINUE
      DO 36 I=1,8
      DO 36 J=1,8
      SUMM(P,Q)=SUMM(P,Q)+(QNT(I+P,J+Q)+QND(I+8,J+8))/SQRT(HD1+HD2)
36  CONTINUE
      IF(SUMM(P,Q).LT.TOTAL)GO TO 52
C
      TOTAL=SUMM(P,Q)
      PMAX=P
      QMAX=Q
      HOLD1=HD1
      HOLD2=HD2
C
      D 5 I=1,8
      D 5 J=1,8
      COR(I+PMAX,J+QMAX)=(QNT(I+PMAX,J+QMAX)+QND(I+8,J+8))/(SQRT(HOLD1
+HOLD2))
5  CONTINUE
52  CONTINUE
17  CONTINUE
84  CONTINUE
85  CONTINUE
C  WRITE(6,62)PMAX,QMAX

```

```
62  FORMAT(1X,*,PMAX=*,I4,*,QMAX=*,I4)
    DO 6 I=1,8
      DO 6 J=1,8
        SUMX=SUMX+FLOAT(I+PMAX)*COR(I+PMAX,J+QMAX)
        SUMY=SUMY+FLOAT(J+QMAX)*COR(I+PMAX,J+QMAX)
        SUMA=SUMA+COR(I+PMAX,J+QMAX)
6   CONTINUE
    XCEN=X-SUMX/SUMA+X-8.0
    YCEN=Y-SUMY/SUMA+Y-8.0
C
    RETURN
    END
```

```
SUBROUTINE CENTRD(X,Y,DATA,N,XCENT,YCENT,THRESH)
  COMPLEX DATA(N,N)
```

```
C
C
C
```

```
  THIS SUBROUTINE PERFORMS THE CORRELATOR THRESHOLDING
```

```
  AMAX=-1.E30
```

```
  DO 10 I=1,N
```

```
  DO 10 J=1,N
```

```
10  AMAX=AMAX1(AMAX,REAL(DATA(I,J)))
```

```
  SUMA=0.
```

```
  SUMX=0.
```

```
  SUMY=0.
```

```
  DO 1 I=1,N
```

```
  DO 1 J=1,N
```

```
  IF(REAL(DATA(I,J)).LT.THRESH+AMAX) DATA(I,J)=0.
```

```
  SUMA=SUMA+REAL(DATA(I,J))
```

```
  SUMX=SUMX+FLOAT(J)*REAL(DATA(I,J))
```

```
1  SUMY=SUMY+FLOAT(I)*REAL(DATA(I,J))
```

```
  XCENT=SUMX/SUMA+X-8.5
```

```
  YCENT=SUMY/SUMA+Y-8.5
```

```
  RETURN
```

```
  END
```



```
SUBROUTINE CHAQUAD(DA A,N)
  COMPLEX DATA(N,N),SAVE1,SAVE2
  N2=N/2
  DO 1 I=1,N2
    DO 1 J=1,N2
      SAVE1=DATA(I,J)
      DATA(I,J)=DATA(12+I,12+J)
      DATA(12+I,12+J)=SAVE1
      SAVE2=DATA(12+I,J)
      DATA(12+I,J)=DATA(I,12+J)
      DATA(I,12+J)=SAVE2
    1 CONTINUE
  RETURN
END
```

```
SUBROUTINE SMOOTH(DATA,SDATA,ALPHA,N,ITERAT)
  COMPLEX DATA(N,N),SDATA(N,N)
  C      THIS ROUTINE SMOOTHS RAW DATA ARRAY DATA USING EXPONENTIAL
  C      SMOOTHING. WEIGHTING FACTOR ALPHA IS USED TO GENERATE THE
  C      SMOOTHED DATA IN ARRAY SDATA. THE PARAMETER ITERATION IS
  C      USED TO DETERMINE THE WEIGHTING FACTOR WHEN FEWER THEN
  C      1/ALPHA ITERATIONS HAVE BEEN DONE.
  A=1./ITERAT
  IF(A.LT.ALPHA) A=ALPHA
  1    DO 3 I=1,N
      DO 3 J=1,N
  3    SDATA(I,J)=A*DATA(I,J)+(1.-A)*SDATA(I,J)
  RETURN
END
```

```

SUBROUTINE SPIN(SIGVAR, M)
  DIMENSION C(64,64),C(5)
  DATA C/.367,.2431,.1353,.1069,.05-1/

```

C
C
C
C
C
C
C
C
C
C
C
C

```

    SET UP SPATIAL NOISE CORRELATION COEFFICIENT MATRIX
    USING SECOND NEAREST NEIGHBOR CORRELATION.

```

```

    C IS THE ARRAY CONTAINING THE NON-ZERO ELEMENTS
    CORRESPONDING TO THE DISTANCES TO NEIGHBORING PIXELS.
    THE ARRAY VALUES ARE EXP(-DISTANCE IN PIXELS)

```

```

    SIGVAR IS THE BACKGROUND VARIANCE

```

```

    P IS THE M**2 BY M**2 CORRELATION MATRIX

```

```

  N=M**2
  DO 30 I=1,N
    DO 30 J=1,N
      P(I,J)=0.0
30  CONTINUE
  DO 36 I=1,N
    P(I,I)=1.
    IF (I.GE.64) GO TO 36
    P(I,I+1)=C(1)
    IF (I.GE.63) GO TO 36
    P(I,I+2)=C(3)
    IF (I.GE.59) GO TO 36
    P(I,I+6)=C(4)
    IF (I.GE.5 GO TO 36
    P(I,I+7)=C(2)
    IF (I.GE.57) GO TO 36
    P(I,I+8)=C(1)
    IF (I.GE.56) GO TO 36
    P(I,I+9)=C(2)
    IF (I.GE.55) GO TO 36
    P(I,I+10)=C(4)
    IF (I.GE.51) GO TO 36
    P(I,I+14)=C(5)
    IF (I.GE.30) GO TO 36
    P(I,I+15)=C(4)
    IF (I.GE.49) GO TO 36
    P(I,I+16)=C(3)
    IF (I.GE.48) GO TO 36
    P(I,I+17)=C(4)
    IF (I.GE.47) GO TO 36
    P(I,I+19)=C(5)
36  CONTINUE
  DO 37 I=1,M
    P(8*I-7,8*I)=0.0
    P(8*I-7,8*I-1)=0.0
    P(8*I-6,8*I)=0.0
    IF (I.GE.8) GO TO 37
    P(8*I,8*I+1)=0.0
    P(8*I,8*I+2)=0.0
    P(8*I-1,8*I+1)=0.0
    P(8*I-7,8*I+7)=0.0

```

```
      R(8*I-7,3*I+1)=0.0  
      R(8*I-6,3*I+1)=0.0  
      IF (1.GE.7) GO TO 37  
      R(8*I,3*I+9)=0.0  
      R(8*I,3*I+10)=0.0  
      R(8*I-1,3*I+9)=0.0  
      IF (1.GE.6) GO TO 37  
      R(8*I,3*I+17)=0.0  
      R(8*I,3*I+18)=0.0  
      R(8*I-1,3*I+17)=0.0  
37    CONTINUE  
      DO 38 I=1,N  
      I=I+1  
      DO 38 J=L,N  
      IF (L.GT.N) GO TO 39  
      R(J,I)=R(I,J)  
38    CONTINUE  
      DO 39 I=1,N  
      DO 39 J=1,N  
39    R(I,J)=SIGMA*R(I,J)  
      RETURN  
      END
```

SUBROUTINE NOISE(W,A)

REAL W(N)

IA=1

```
1 DO 2 I=1,N
  CALL GAUSS(IA,IY,VAL)
  W(I)=VAL
  IA=IY
2 CONTINUE
  RETURN
END
```

SUBROUTINE SHIFT(DATA,N,XSHIFT,YSHIFT)
 COMPLEX DATA(N,N),TEMP1,TEMP2,TEMP3,TEMP4,FX,FXC,FY,FYC
 THIS ROUTINE IMPLEMENTS A SPATIAL PHASE SHIFT
 IN THE FREQUENCY DOMAIN. THE ARRAY DATA IS ASSUMED TO BE THE
 NXN ARRAY OF FOURIER TRANSFORM COMPONENTS AS GENERATED BY THE
 TRANSFORM ROUTINE FOURT. FOR EXAMPLE, FOR A 6X6 ARRAY DATA

```

-----
[ X0 Y0 [ X1 Y0 [ X2 Y0 [ X3 Y0 [X2* Y0 [X1* Y0 [
-----
[ X0 Y1 [ X1 Y1 [ X2 Y1 [ X3 Y1 [X2* Y1 [X1* Y1 [
-----
[ X0 Y2 [ X1 Y2 [ X2 Y2 [ X3 Y2 [X2* Y2 [X1* Y2 [
-----
[ X0 Y3 [ X1 Y3 [ X2 Y3 [ X3 Y3 [X2* Y3 [X1* Y3 [
-----
[ X0 Y2* [ X1 Y2* [ X2 Y2* [ X3 Y2* [X2* Y2* [X1* Y2* [
-----
[ X0 Y1* [ X1 Y1* [ X2 Y1* [ X3 Y1* [X2* Y1* [X1* Y1* [
-----

```

PHASE SHIFTING IS IMPLEMENTED BY MULTIPLYING THE
 FOURIER TRANSFORM COMPONENTS BY
 $\exp(j \cdot 2 \cdot \pi (FX \cdot XSHIFT + FY \cdot YSHIFT))$

XSHIFT AND YSHIFT ARE THE SHIFTS IN THE X AND Y COORDINATE
 DIRECTIONS.

```

7  PI=3.141592654
   DEM=FLOAT(N)
   NCENT=N/2+1
   DO 1 I=1,NCENT
     DO 1 J=1,NCENT
       FX=CMPLX(0.,-2.*PI*(J-1)*XSHIFT/DEM)
       FY=CMPLX(0.,-2.*PI*(I-1)*YSHIFT/DEM)
       FXC=CONJG(FX)
       FYC=CONJG(FY)
       TEMP1=DATA(I,J)
       DATA(I,J)=TEMP1*CEXP(FX+FY)
       IF(I.EQ.1) GO TO 10
       IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 20
       TEMP2=DATA(I,N+2-J)
       TEMP3=DATA(N+2-I,J)
       TEMP4=DATA(N+2-I,N+2-J)
       DATA(I,N+2-J)=TEMP2*CEXP(FXC+FY)
       DATA(N+2-I,J)=TEMP3*CEXP(FX+FYC)
       DATA(N+2-I,N+2-J)=TEMP4*CEXP(FXC+FYC)
       GO TO 1
10  IF(J.EQ.1.OR.J.EQ.NCENT) GO TO 1
     TEMP2=DATA(I,N+2-J)
     DATA(I,N+2-J)=TEMP2*CEXP(FXC+FY)
     GO TO 1
20  TEMP3=DATA(N+2-I,J)
     DATA(N+2-I,J)=TEMP3*CEXP(FX+FYC)
1  CONTINUE
   RETURN

```

UBROUTINE FOURT(DATA,NN,NDIM,ISIGN,IFORM,WORK)
FOR INFORMATION CONTACT MR. MARK HALLER 4950/ADD/5624

THE COOLEY-TUKEY FAST FOURIER TRANSFORM IN USASI BASIC FORTRAN

TRANSFORM(K1,K2,...) = SUM(DATA(J1,J2,...)*EXP(ISIGN*2*PI*SQRT(-1)
+((J1-1)*(K1-1)/NN(1)+(J2-1)*(K2-1)/NN(2)+...))), SUMMED FOR ALL
J1, K1 BETWEEN 1 AND NN(1), J2, K2 BETWEEN 1 AND NN(2), ETC.
THERE IS NO LIMIT TO THE NUMBER OF SUBSCRIPTS. DATA IS A
MULTIDIMENSIONAL COMPLEX ARRAY WHOSE REAL AND IMAGINARY
PARTS ARE ADJACENT IN STORAGE, SUCH AS FORTRAN IV PLACES THEM.
IF ALL IMAGINARY PARTS ARE ZERO (DATA ARE DISGUISED REAL), SET
IFORM TO ZERO TO CUT THE RUNNING TIME BY UP TO FORTY PERCENT.
OTHERWISE, IFORM = +1. THE LENGTHS OF ALL DIMENSIONS ARE
STORED IN ARRAY NN, OF LENGTH NDIM. THEY MAY BE ANY POSITIVE
INTEGERS, THO THE PROGRAM RUNS FASTER ON COMPOSITE INTEGERS, AND
ESPECIALLY FAST ON NUMBERS RICH IN FACTORS OF TWO. ISIGN IS +1
OR -1. IF A -1 TRANSFORM IS FOLLOWED BY A +1 ONE (OR A +1
BY A -1) THE ORIGINAL DATA REAPPEAR, MULTIPLIED BY NTOT (=NN(1)*
NN(2)*...). TRANSFORM VALUES ARE ALWAYS COMPLEX, AND ARE RETURNED
IN ARRAY DATA, REPLACING THE INPUT. IN ADDITION, IF ALL
DIMENSIONS ARE NOT POWERS OF TWO, ARRAY WORK MUST BE SUPPLIED,
COMPLEX OF LENGTH EQUAL TO THE LARGEST NON 2**K DIMENSION.
OTHERWISE, REPLACE WORK BY ZERO IN THE CALLING SEQUENCE.
NORMAL FORTRAN DATA ORDERING IS EXPECTED, FIRST SUBSCRIPT VARYING
FASTEST. ALL SUBSCRIPTS BEGIN AT ONE.

RUNNING TIME IS MUCH SHORTER THAN THE NAIVE NTOT**2, BEING
GIVEN BY THE FOLLOWING FORMULA. DECOMPOSE NTOT INTO
2**K2 * 3**K3 * 5**K5 * LET SUM2 = 2*K2, SUMF = 3*K3 + 5*K5
+ ... AND NF = K3 + K5 + THE TIME TAKEN BY A MUL I-
DIMENSIONAL TRANSFORM ON THESE NTOT DATA IS $T = T_0 + NTOT * (T_1 +$
 $T_2 * SUM2 + T_3 * SUMF + T_4 * NF)$. ON THE CDC 3300 (FLOATING POINT ADD TIME
OF SIX MICROSECONDS), $T = 3000 + NTOT * (500 + 43 * SUM2 + 68 * SUMF +$
 $320 * NF)$ MICROSECONDS ON COMPLEX DATA. IN ADDITION, THE
ACCURACY IS GREATLY IMPROVED, AS THE RMS RELATIVE ERROR IS 9999999
BOUNDED BY $3 * 2 ** (-B) * SUM(FACTOR(J) ** 1.5)$, WHERE B IS THE NUMBER
OF BITS IN THE FLOATING POINT FRACTION AND FACTOR(J) ARE THE
PRIME FACTORS OF NTOT.

PROGRAM BY NORMAN BRENNER FROM THE BASIC PROGRAM BY CHARLES
RADEP. RALPH ALTER SUGGESTED THE IDEA FOR THE DIGIT REVERSAL.
MIT LINCOLN LABORATORY, AUGUST 1967. THIS IS THE FASTEST AND MOST
VERSATILE VERSION OF THE FFT KNOWN TO THE AUTHOR. SHORTER PRO-
GRAMS FOUR1 AND FOUR2 RESTRICT DIMENSION LENGTHS TO POWERS OF TWO.
SEE-- IEEE AUDIO TRANSACTIONS (JUNE 1967), SPECIAL ISSUE ON FFT.

THE DISCRETE FOURIER TRANSFORM PLACES THREE RESTRICTIONS UPON THE
DATA.

1. THE NUMBER OF INPUT DATA AND THE NUMBER OF TRANSFORM VALUES
MUST BE THE SAME.
2. BOTH THE INPUT DATA AND THE TRANSFORM VALUES MUST REPRESENT
EQUISPACED POINTS IN THEIR RESPECTIVE DOMAINS OF TIME AND
FREQUENCY. CALLING THESE SPACINGS DELTAT AND DELTAF, IT MUST BE
TRUE THAT $DELTAF = 2 * PI / (NN(1) * DELTAT)$. OF COURSE, DELTAT NEED NOT
BE THE SAME FOR EVERY DIMENSION.

3. CONCEPTUALLY AT LEAST, THE INPUT DATA AND THE TRANSFORM OUTPUT REPRESENT SINGLE CYCLES OF PERIODIC FUNCTIONS.

EXAMPLE 1. THREE-DIMENSIONAL FORWARD FOURIER TRANSFORM OF A COMPLEX ARRAY DIMENSIONED 32 BY 25 BY 13 IN FORTRAN IV.

DIMENSION DATA(32,25,13),WORK(53),NV(3)

COMPLEX DATA

DATA NV/32,25,13/

DO 1 I=1,32

DO 1 J=1,25

DO 1 K=1,13

1 DATA(I,J,K)=COMPLEX VALUE

CALL FOURT(DATA,NV,3,-1,1,WORK)

EXAMPLE 2. ONE-DIMENSIONAL FORWARD TRANSFORM OF A REAL ARRAY OF LENGTH 64 IN FORTRAN II.

DIMENSION DATA(2,64)

DO 2 I=1,64

DATA(1,I)=REAL PART

2 DATA(2,I)=0.

CALL FOURT(DATA,64,1,-1,0,0)

DIMENSION DATA(1),NV(1),IFACT(32),WORK(1)

CDC 6600 INITIALIZATION

WRITE(6,5000)

WR=0.

WI=0.

WSTPR=0.

WSTPI=0.

FWOPT=6.283185307

IF(NDIM-1)920,1,1

1 NTOT=2

DO 2 IDIM=1,NDIM

IF(NV(IDIM))920,920,2

2 NTOT=NTOT*NV(IDIM)

MAIN LOOP FOR EACH DIMENSION

NP1=2

DO 310 IDIM=1,NDIM

M=NV(IDIM)

NP2=NP1*

IF(M-1)920,900,5

C FACTOR M

5 M=M

NTWO=NP1

F=1

IDIV=2

10 IQUOT=M/IDIV

IFEM=M-IDIV*IQUOT

IF(IQUOT-IDIV)50,11,11

11 IF(IEM)20,12,20

12 NTWO=NTWO+NTWO

M=IQUOT

GO TO 10


```

20  IDIV=3
30  IQUOT=M/IDIV
    IREM=M-IDIV*IQUOT
    IF(IQUOT-IDIV)60,31,31
31  IF(IREM)40,32,40
32  IFACT(IF)=IDIV
    IF=IF+1
    M=IQUOT
    GO TO 30
40  IDIV=IDIV+2
    GO TO 30
50  IF(IREM)60,51,60
51  NTWO=NTWO+NTWO
    GO TO 70
60  IFACT(IF)=M
C
C  SEPARATE FOUR CASES--
C    1. COMPLEX TRANSFORM OR REAL TRANSFORM FOR THE 4TH, 5TH, ETC.
C      DIMENSIONS.
C    2. REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION. METHOD--
C      TRANSFORM HALF THE DATA, SUPPLYING THE OTHER HALF BY CON-
C      JUGATE SYMMETRY.
C      JUGATE SYMMETRY.
C    3. REAL TRANSFORM FOR THE 1ST DIMENSION, N ODD. METHOD--
C      TRANSFORM HALF THE DATA AT EACH STAGE, SUPPLYING THE OTHER
C      HALF BY CONJUGATE SYMMETRY.
C    4. REAL TRANSFORM FOR THE 1ST DIMENSION, N EVEN. METHOD--
C      TRANSFORM A COMPLEX ARRAY OF LENGTH N/2 WHOSE REAL PARTS
C      ARE THE EVEN NUMBERED REAL VALUES AND WHOSE IMAGINARY PARTS
C      ARE THE ODD NUMBERED REAL VALUES. SEPARATE AND SUPPLY
C      THE SECOND HALF BY CONJUGATE SYMMETRY.
C
70  NON2=NP1*(NP2/NTWO)
    ICASE=1
    IF(IDIM-4)71,90,70
71  IF(IFORM)72,72,90
72  ICASE=2
    IF(IDIM-1)73,73,90
73  ICASE=3
    IF(NTWO-NP1)90,90,74
74  ICASE=4
    NTWO=NTWO/2
    N=N/2
    NP2=NP2/2
    NTOT=NTOT/2
    I=3
    DO 90 J=2,NTOT
    DATA(J)=DATA(I)
90  I=I+2
99  IIRNG=NP1
    IF(ICASE-2)100,95,100
95  IIRAG=NP0*(1+NPREV/2)
C
C  SHUFFLE ON THE FACTORS OF TWO IN N. AS THE SHUFFLING
C  CAN BE DONE BY SIMPLE INTERCHANGE, NO WORKING ARRAY IS NEEDED
C
100 IF(NTWO-NP1)600,600,110

```

```

110 NP2HF=NP2/2
    J=1
    DO 150 I2=1, NP2, NON2
    IF (J-I2) 120, 130, 130
120 I1MAX=I2+NON2-2
    DO 125 I1=I2, I1MAX, 2
    DO 125 I3=I1, I1+1, NP2
    J3=J+I3-2
    TEMPR=DATA(I3)
    TEMPI=DATA(I3+1)
    DATA(I3)=DATA(J3)
    DATA(I3+1)=DATA(J3+1)
    DATA(J3)=TEMPR
125 DATA(J3+1)=TEMPI
130 M=NP2HF
140 IF (J-M) 150, 150, 145
145 J=J-M
    M=M/2
    IF (M-NON2) 150, 140, 140
150 J=J+M
C
C MAIN LOOP FOR FACTORS OF TWO. PERFORM FOURIER TRANSFORMS OF
C LENGTH FOUR, WITH ONE OF LENGTH TWO IF NEEDED. THE TWIDDLE FACTOR
C W=EXP(ISIGN*2*PI*SQRT(-1)*M/(4*MMAX)). CHECK FOR W=ISIGN*SQRT(-1)
C AND REPEAT FOR W=ISIGN*SQRT(-1)*CONJUGATE(W).
C
NON2I=NON2+M/2
IPAR=NTWO/NP1
310 IF (IPAR-2) 350, 330, 320
320 IPAR=IPAR/4
    GO TO 310
330 DO 340 I1=1, I1 MAX, 2
    DO 340 J3=I1, NON2, NP1
        DO 340 K1=J3, NTOT, NON2I
        K2=K1+NON2
        TEMPR=DATA(K2)
        TEMPI=DATA(K2+1)
        DATA(K2)=DATA(K1)-TEMPI
        DATA(K2+1)=DATA(K1+1)-TEMPI
        DATA(K1)=DATA(K1)+TEMPR
        DATA(K1+1)=DATA(K1+1)+TEMPI
340 DATA(K1+1)=DATA(K1+1)+TEMPI
350 MMAX=NON2
360 IF (MMAX-NP2HF) 370, 600, 600
370 LMAX=MAX0(NON2I, MMAX/2)
    IF (MMAX-NON2) 405, 405, 360
380 THETA=-TWOPI*FLOAT(NON2)/FLOAT(4*MMAX)
    IF (ISIGN) 400, 390, 380
390 THETA=-THETA
400 WR=COS(THETA)
    WI=SIN(THETA)
    WSTPR=-2.*WI*WI
    WSTPI=2.*WR*WI
405 DO 570 L=NON2, LMAX, NON2I
    M=L
    IF (MMAX-NON2) 420, 420, 410
410 W2R=WR*WR-WI*WI
    W2I=2.*WR*WI

```

```

W3R=W2R*WR-W2I*WI
W3I=W2R*WI+W2I*WR
420 DO 530 I1=1,11 NG,2
DO 530 J3=11,NOV2,N
KMIN=J3+IPA*M
IF(MMAX-NON2)430,430,440
430 KMIN=J3
440 KDIF=IPA*M*MMAX
450 KSTEP=4*KDIF
DO 520 K1=KMIN,1107,KSTEP
K2=K1+KDIF
K3=K2+KDIF
K4=K3+KDIF
IF(MMAX-NON2)460,460,480
460 U1R=DATA(K1)+DATA(K2)
U1I=DATA(K1+1)+DATA(K2+1)
U2R=DATA(K3)+DATA(K4)
U2I=DATA(K3+1)+DATA(K4+1)
U3R=DATA(K1)-DATA(K2)
U3I=DATA(K1+1)-DATA(K2+1)
IF(ISIG)470,475,475
470 U4R=DATA(K3+1)-DATA(K4+1)
U4I=DATA(K4)-DATA(K3)
GO TO 510
475 U4R=DATA(K4+1)-DATA(K3+1)
U4I=DATA(K3)-DATA(K4)
GO TO 510
480 T2R=W2R+DATA(K2)-W2I+DATA(K2+1)
T2I=W2R+DATA(K2+1)+W2I+DATA(K2)
T3R=WR+DATA(K3)-WI+DATA(K3+1)
T3I=WR+DATA(K3+1)+WI+DATA(K3)
T4R=W3R+DATA(K4)-W3I+DATA(K4+1)
T4I=W3R+DATA(K4+1)+W3I+DATA(K4)
U1R=DATA(K1)+T2R
U1I=DATA(K1+1)+T2I
U2R=T3R+T4R
U2I=T3I+T4I
U3R=DATA(K1)-T2R
U3I=DATA(K1+1)-T2I
IF(I-IGL)490,500,500
490 U4R=T3I-T4I
U4I=T4R-T3R
GO TO 510
500 U4R=T4I-T3I
U4I=T3R-T4R
510 DATA(K1)=U1R+U2R
DATA(K1+1)=U1I+U2I
DATA(K2)=U3R+U4R
DATA(K2+1)=U3I+U4I
DATA(K3)=U1R-U2R
DATA(K3+1)=U1I-U2I
DATA(K4)=U3R-U4R
520 DATA(K4+1)=U3I-U4I
KMIN=4*(KMIN-J3)+J3
KDIF=KSTEP
KDIF=KSTEP
IF(KDIF-LP2)450,530,530

```

```

530  CONTINUE
      M=MMAX-M
      IF (ISIGN) 540,550,550
540  TEMPR=WR
      WR=-WI
      WI=-TEMPR
      GO TO 560
550  TEMPR=WR
      WR=WI
      WI=TEMPR
560  IF (M-LMAX) 565,565,410
565  TEMPR=WR
      WR=WR*WSTPR-WI*WSTPI+WP
570  WI=WI*WSTPR+TEMPR*WSTPI+WI
      IPAR=3-IPAR
      MMAX=MMAX+MMAX
      GO TO 340

C
C   MAIN LOOP FOR FACTORS NOT EQUAL TO TWO. APPLY THE TWIDDLE FACTOR
C    $W = \exp(i \text{SIGN} * 2 * \pi * \text{SQRT}(-1) * (J2-1) * (J1-J2) / (NP2 * IFP1))$ , THEN
C   PERFORM A FOURIER TRANSFORM OF LENGTH IFACT(IF), MAKING USE OF
C   CONJUGATE SYMMETRIES.
C
600  IF (NTWO-NP2) 605,700,700
605  IFP1=NON2
      IF=1
      NP1HF=NP1/2
610  IFP2=IFP1/IFACT(IF)
      J1RNG=NP2
      IF (ICASE-3) 612,611,612
611  J1RNG=(NP2+IFP1)/2
      J2STP=NP2/IFACT(IF)
      J1RG2=(J2STP+IFP2)/2
612  J2MIN=1+IFP2
      IF (IFP1-NP2) 615,640,640
615  DO 635 J2=J2MIN,IFP1,IFP2
      THETA=-TWOPI*FLOAT(J2-1)/FLOAT(NP2)
      IF (ISIGN) 625,620,620
620  THETA=-THETA
625  SINTH=SIN(THETA/2.)
      WSTPR=-2.*SINTH*SINTH
      WSTPI=SI(THETA)
      WR=WSTPR+1.
      WI=WSTPI
      J1MIN=J2+IFP1
      DO 635 J1=J1MIN,J1RNG,IFP1
          I1MAX=J1+I1RG-2
          DO 630 I1=J1,I1MAX,2
              DO 630 I3=I1,NTWO,NP2
                  J3MAX=I3+IFP2-NP1
                  DO 630 J3=I3,J3MAX,NP1
                      TEMPR=DATA A(J3)
                      DATA(J3)=DATA(J3)*WR-DATA(J3+1)*WI
630  DATA(J3+1)=TEMPR*WI+DATA(J3+1)*J2
                      TEMPR=WR
                      WR=WR*WSTPR-WI*WSTPI+WR
635  WI=TEMPR*WSTPI+WI*WSTPR+WI

```

```

640 THETA=-TWOPI/FLCAT(IFACT(IF))
    IF(I.IGN)650,645,640
645 THETA=-THETA
650 SINTH=SIN(THETA/2.)
    WSTPR=-2.*SINTH*SIN H
    WSTPI=SIN(THETA)
    KSTEP=2*N/IFACT(IF)
    KRANG=KSTEP*(IFACT(IF)/2)+1
    DO 660 I1=1,I1=NG,2
    DO 660 I3=I1,NTO,NP2
    DO 660 KMIN=1,KRANG,KSTEP
    J1MAX=I3+J1RANG-IFP1
    DO 660 J1=I3,J1MAX,IFP1
    J3MAX=J1+IFP2-NP1
    DO 660 J3=J1,J3MAX,NP1
    J2MAX=J3+IFP1-IFP2
    K=KMIN+(J3-J1+(J1-I3)/IFACT(IF))/NP1HF
    IF(KMIN-1)655,655,665
655 SUMR=0.
    SUMI=0.
    DO 660 J2=J3,J2MAX,IFP2
    SUMR=SUMR+DATA(J2)
660 SUMI=SUMI+DATA(J2+1)
    WORK(K)=SUMR
    WORK(K+1)=SUMI
    GO TO 690
665 KCONJ=K+2*(N-KMIN+1)
    J2=J2MAX
    SUMR=DATA(J2)
    SUMI=DATA(J2+1)
    OLDSP=0.
    OLDSI=0.
    J2=J2-IFP2
670 TEMPR=SUMR
    TEMPI=SUMI
    SUMR=TWOHR*SUMR-OLDSP+DATA(J2)
    SUMI=TWOHR*SUMI-OLDSI+DATA(J2+1)
    OLDSP=TEMPR
    OLDSI=TEMPI
    J2=J2-IFP2
    IF(J2-J3)675,675,670
675 TEMPR=WR*SUMR-OLDSP+DATA(J2)
    TEMPI=WI*SUMI
    WORK(K)=TEMPR-TEMPI
    WORK(KCONJ)=TEMPR+TEMPI
    TEMPR=WR*SUMR-OLDSI+DATA(J2+1)
    TEMPI=WI*SUMI
    WORK(K+1)=TEMPR+TEMPI
    WORK(KCONJ+1)=TEMPR-TEMPI
680 CONTINUE
    IF(KMIN-1)685,685,686
685 WR=WSTPR+1.
    WI=WSTPI
    GO TO 690
686 TEMPR=WR
    WR=WR*WSTPR-WI*WSTPI+WR
    WI=TEMPR*WSTPI+WI*WSTPR+WI

```

```

650  TWOWP=WP+WR
    IF(ICASE-3)642,651,652
651  IF(IFP1-NP2)655,652,652
652  K=1
    I2MAX=I3+NP2-NP1
    DO 653 I2=I3,I2MAX,NP1
    DATA(I2)=WORK(K)
    DATA(I2+1)=WORK(K+1)
653  K=K+2
    GO TO 650
C
C   COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N ODD, BY CON-
C   JUGATE SYMMETRIES AT EACH STAGE.
C
655  J3MAX=I3+IFP2-NP1
    DO 657 J3=I3,J3MAX,NP1
    J2MAX=J3+NP2-J2STP
    DO 657 J2=J3,J2MAX,J2STP
    J1MAX=J2+J1SG2-IFP2
    J1CMJ=J3+J2MAX+J2STP-J2
    DO 657 J1=J2,J1MAX,IFP2
    K=1+J1-I3
    DATA(J1)=WORK(K)
    DATA(J1+1)=WORK(K+1)
    IF(J1-J2)657,657,656
656  DATA(J1CMJ)=WORK(K)
    DATA(J1CMJ+1)=-WORK(K+1)
657  J1CMJ=J1CMJ-IFP2
658  CONTINUE
    IF=IF+1
    IFP1=IFP2
    IF(IFP1-NP1)700,700,610
C
C   COMPLETE A REAL TRANSFORM IN THE 1ST DIMENSION, N EVEN, BY CON-
C   JUGATE SYMMETRIES.
C
700  GO TO (900,400,500,701),ICASE
701  NHALF=N/2
    THETA=-TWOPY/FLOAT(N)
    IF(1-IGN)703,702,702
702  THETA=-THETA
703  SINTH=SIN(THETA/2.)
    WSTPR=-2.*SINTH*SINTH
    WSTPI=SIN(THETA)
    WR=WSTPR+1.
    WI=WSTPI
    IMI=3
    JMIN=2*NHALF-1
    GO TO 725
710  J=JMIN
    DO 720 I=IMI,NTOT,NP2
    SUMR=(DATA(I)+DATA(J))/2.
    SUMI=(DATA(I+1)+DATA(J+1))/2.
    DIFF=(DATA(I)-DATA(J))/2.
    DIFI=(DATA(I+1)-DATA(J+1))/2.
    TEMPR=WR+SUMI+WI*DIFF

```

```

      TEMPI=WI*SUMI-UI*DIFF
      DATA(I)=SUMI*TEMP
      DATA(I+1)=DIFFI*TEMP
      DATA(J)=SUMI-TEMP
      DATA(J+1)=-DIFFI*TEMP
720   J=J+NP2
      IMIN=IMIN+2
      JMIN=JMIN-2
      TEMPR=WR
      WS=WR*WSTPR-WI*WSTPI+WR
      WI=TEMPR*WSTPI+WI*WSTPR+WI
725   IF(IMIN-JMIN)710,730,740
730   IF(I-IGN)731,740,740
731   DO 735 I=IMIN,NTOT,NP2
735   DATA(I+1)=-DATA(I+1)
740   NP2=NP2+NP2
      NTOT=NTOT+NTOT
      J=NTOT+1
      IMAX=NTOT/2+1
745   IMIN=IMAX-2*HALF
      I=IMIN
      GO TO 755
750   DATA(J)=DATA(I)
      DATA(J+1)=-DATA(I+1)
755   I=I+2
      J=J-2
      IF(I-IMAX)750,760,760
760   DATA(J)=DATA(IMIN)-DATA(IMIN+1)
      DATA(J+1)=0.
      IF(I-J)770,780,780
765   DATA(J)=DATA(I)
      DATA(J+1)=DATA(I+1)
770   I=I-2
      J=J-2
      IF(I-IMIN)775,775,765
775   DATA(J)=DATA(IMIN)+DATA(IMIN+1)
      DATA(J+1)=0.
      IMAX=IMIN
      GO TO 745
780   DATA(1)=DATA(1)+DATA(2)
      DATA(2)=0.
      GO TO 900
C
C   COMPLETE A REAL TRANSFORM FOR THE 2ND OR 3RD DIMENSION BY
C   CONJUGATE SYMMETRIES.
C
900   IF(I1RNG-NP1)905,900,900
905   DO 960 I3=1,NTOT,NP2
      I2MAX=I3+NP2-NP1
      DO 960 I2=I3,I2MAX,NP1
      IMIN=I2+I1*G
      IMAX=I2+NP1-2
      JMAX=2*I3+NP1-IMIN
      IF(I2-I3)820,820,810
810   JMAX=JMAX+NP2
820   IF(IDIN-2)850,850,830
830   J=JMAX+NP0

```

```
      DO 840 I=IMIN,IMAX,2
      DATA(I)=DATA(J)
      DATA(I+1)=-DATA(J+1)
840   J=J-2
850   J=JMAX
      DO 860 I=IMIN,IMAX,NP0
      DATA(I)=DATA(J)
      DATA(I+1)=-DATA(J+1)
860   J=J-NP0
      C
      C   END OF LOOP ON EACH DIMENSION
      C
800   NP0=NP1
      NP1=NP2
810   NPREV=N
820   RETURN
      END
```

CRITY DETAILS DIAGNOSIS OF PROBLEM

```
DATA      ARRAY REFERENCE OUTSIDE DIMENSION BOUNDS.
DATA      ARRAY REFERENCE OUTSIDE DIMENSION BOUNDS.
```



```

C      SUBROUTINE GAUSS(IA,IY,VAL)
C      THIS ROUTINE CALCULATE A GAUSSIAN DISTRIBUTED RANDOM VARIABLE
C      VAL, WITH MEAN=0. AND STANDARD DEVIATION=1.
C
C      IA IS INITIALIZED BEFORE FIRST CALL TO ANY ODD INTEGER LESS THEN
C      10 DIGITS IN LENGTH.
C      IY IS GENERATED AND SHOULD BE USED FOR IA ON THE NEXT CALL 0
C      THIS ROUTINE.
C      VAL=0.
C      DO 1 I=1,12
C      X=RAMF(DUM)
C      IA=IY
C      VAL=VAL+X
1     CONTINUE
C      VAL=VAL-6.
C      RETURN
C      END

```

```

SUBROUTINE INVERT(A,N,D,IER)
IMPLICIT REAL (A-H,O-Z)
    DIMENSION A(1),B(1)
    REAL L(128),M(128)
    NSQ=N*N
    DO 1000 I=1,N*Q
1000 H(I)=A(I)
    D=1.0
    KK=-N
    DO 30 K=1,
    KK=KK+
    L(K)=K
    M(K)=K
    KK=K+K
    BIGA=A(KK)
    DO 20 J=K,
    IZ=N*(J-1)
    DO 20 I=K,
    IJ=IZ+I
10 IF(ABS(BIGA)-ABS(A(IJ))) 15,20,20
15 BIGA=A(IJ)
    L(K)=I
    M(K)=J
20 CONTINUE
    J=L(K)
    IF(J-K) 35,35,25
25 KI=K-M
    DO 30 I=1,
    KI=KI+
    HOLD=-A(KI)
    JI=KI-K+J
    A(KI)=A(JI)
30 A(JI)=HOLD
35 I=M(K)
    IF(I-K) 45,45,38
38 JP=N*(I-1)
    DO 40 J=1,
    JK=K+J
    JI=JP+J
    HOLD=-A(JK)
    A(JK)=A(JI)
40 A(JI)=HOLD
45 IF(BIGA) 45,46,48
46 D=0.0
    IER=129
    GO TO 150
48 DO 55 I=1,
    IF(I-K) 50,55,50
50 IK=K+I
    A(IK)=A(IK)/(-BIGA)
55 CONTINUE
    DO 65 I=1,
    IK=K+I
    IJ=I-
    DO 65 J=1,
60 IF(J-K) 62,65,62
62 KJ=IJ-I+K
    
```

```
      A(IJ)=A(IK)*A(KJ)+A(IJ)
65      CONTINUE
      KJ=K-1
      DO 75 J=1,
      KJ=KJ+
      IF(J-K) 70,75,70
70      A(KJ)=A(KJ)/BIGA
75      CONTINUE
      D=D*BIGA
      A(KK)=1.0/BIGA
80      CONTINUE
      K=
      K=(K-1)
      IF(K) 150,150,105
105      I=L(K)
      IF(I-K) 120,120,100
100      JQ=I*(K-1)
      JF=I*(I-1)
      DO 110 J=1,
      JK=JQ+J
      HOLD=A(JK)
      JI=JF+J
      A(JK)=-A(JI)
110      A(JI)=HOLD
120      J=M(K)
      IF(J-K) 100,100,125
125      KI=K-1
      DO 130 I=1,
      KI=KI+
      HOLD=A(KI)
      JI=KI-K+J
      A(KI)=-A(JI)
130      A(JI)=HOLD
      GO TO 100
150      DO 1002 I=1,N50
      SAVE=A(I)
      A(I)=B(I)
      B(I)=SAVE
1002      CONTINUE
      RETURN
      END
```

```
      SUBROUTINE MULT(A,B,L,M,N,C)
      REAL A(L,M),B(M,N),C(L,N)
      DO 300 I=1,L
        DO 200 J=1,N
          C(I,J)=0.
          DO 100 INDEX=1,M
            C(I,J)=C(I,J)+A(I,INDEX)*B(INDEX,J)
          CONTINUE
        CONTINUE
      CONTINUE
      RETURN
      END
```

10
200
300

```
      SUBROUTINE CHOLY(A,N)
C      THIS ROUTINE DETERMINES THE LOWER TRIANGULAR CHOLESKY SQUARE-
C      ROOT OF AN NxN MATRIX.
C      A IS THE INPUT MATRIX AND A IS THE CHOLESKY SQUARE-ROOT MATRIX.
      DIMENSION A(N,N)
      DO 123 I=1,N
        DO 123 J=1,N
          IF(ABS(A(I,J)).GT.1.E-25) GO TO 124
123      CONTINUE
        DO 125 I=1,N
          DO 125 J=1,N
125          A(I,J)=0.
        RETURN
124      A(I,1)=SQRT(A(I,1))
        DO 5 I=2,N
          IM1=I-1
          DO 3 J=1,IM1
            JM1=J-1
            SUM=0.
            IF(J.EQ.1) GO TO 3
            DO 2 K=1,JM1
2              SUM=SUM+A(I,K)*A(J,K)
3              A(I,J)=(A(I,J)-SUM)/A(J,J)
            SUM=0.
            DO 4 K=1,IM1
4              SUM=SUM+A(I,K)**2
5              A(I,I)=SQRT(A(I,I)-SUM)
        DO 6 I=1,N
          IP1=I+1
          DO 6 J=IP1,N
6            A(I,J)=0.
        RETURN
      END
```

```
SUBROUTINE PLTA(TIMPL,A,B,NFRAMES,NIM,I,NAME)
```

```
C  
C
```

```
  INTEGER I  
  REAL B(NIM),A(NIM),TIMPL(NIM)  
  DIMENSION NAME(45)
```

```
C  
C  
C  
C  
C  
C
```

```
  THIS SUBROUTINE PERFORMS THE PLOTTING CALLS REQUIRED  
  TO PLOT THE FILTER ESTIMATED RMS ERRORS VS. THE ACTUAL  
  ERRORS
```

```
  N2=NFRAMES*2  
  CALL SCALE(TIMPL,6.,2,1)  
  B(N2+1)=-.1  
  B(N2+2)=1.25  
  CALL AXIS(0.,0.,9HTIME(SEC),-3,5.,0.,TIMPL(N2+1),TIMPL(N2+2))  
  CALL AXIS(0.,0.,13ERROR(PIXELS),13,4.,90.,B(N2+1),B(N2+2))  
  *)  
  CALL LINE(TIMPL,B,N2,1,0,3)  
  CALL PLOT(0.,0.,3)  
  A(N2+1)=B(N2+1)  
  A(N2+2)=B(N2+2)  
  CALL LINE(TIMPL,A,12,1,0,0)  
  CALL PLOT(-1.,-1.,-3)  
  CALL PLOT(8.,0.,2)  
  CALL PLOT(9.,6.,2)  
  CALL PLOT(0.,6.,2)  
  CALL PLOT(0.,0.,2)  
  CALL SYMBOL(1.0,5.5,0.15,NAME(I),0.,36)  
  CALL PLOT(0.,0.,3)  
  CALL PLOT(13.,1.,-3)
```

```
C  
C
```

```
  RETURN  
  END
```

```

SUBROUTINE PLTR(TIME,A,B,C,NIM,NFRAMES,I,NAME,
+K,NIN,PTA,PTH,PTC,PID)

```

C

```

INTEGER NIM,NIN,NFRAMES,I,K,J,N2,COUNT
REAL TIME(NIM),A(4,NFRAMES),B(4,NFRAMES),C(4,NFRAMES)
REAL PTA(NIN),PTB(NIN),PTC(NIN),PID(NIN)
DIMENSION NAME(45)

```

C

C

C

C

C

```

THIS SUBROUTINE PLOTS THE MEAN ERRORS (+/-) ONE SIGMA
USING THE DATA FROM THE FIRST ROUTINE. THE NAMES
FOR THE ARRAYS WERE PREVIOUSLY STORED IN ARRAY NAME.

```

```

N2=NFRAMES*2

```

```

COUNT=1

```

C

C

```

DO 20 J=1,NFRAMES

```

```

    PTA(COUNT)=B(1,J)

```

```

    PTA(COUNT+1)=A(1,J)

```

```

    PTB(J)=A(1,J)

```

```

    PTC(J)=B(1,J)

```

```

    PID(J)=C(1,J)

```

```

    COUNT=COUNT+2

```

20

```

CONTINUE

```

```

CALL SCALE(PTA,4.,N2,1)

```

```

PTB(NFRAMES+1)=PTA(N2+1)

```

```

PTC(NFRAMES+1)=PTA(N2+1)

```

```

PID(NFRAMES+1)=PTA(N2+1)

```

```

PTH(NFRAMES+2)=PTA(N2+2)

```

```

PTC(NFRAMES+2)=PTA(N2+2)

```

```

PID(NFRAMES+2)=PTA(N2+2)

```

```

CALL SCALE(TIME,6.,NFRAMES,1)

```

```

CALL AXIS(0.,0.,8*TIME(SEC),-9.5,0.,TIME(NFRAMES+1),
+TIME(NFRAMES+2))

```

```

CALL AXIS(0.,0.,13*HEPPO(Pixels),13.4,90.,PTA(N2+1),PTA(N2+2))

```

```

CALL LINE(TIME,PTB,NFRAMES,1,0,0)

```

```

CALL PLOT(0.,0.,3)

```

```

CALL LINE(TIME,PTC,NFRAMES,1,0,1)

```

```

CALL PLOT(0.,0.,3)

```

```

CALL LINE(TIME,PID,NFRAMES,1,0,0)

```

```

CALL PLOT(-1.,-1.,-3)

```

C

```

CALL PLOT(8.,0.,2)

```

```

CALL PLOT(8.,4.,2)

```

```

CALL PLOT(0.,4.,2)

```

```

CALL PLOT(0.,0.,2)

```

```

IF(K.GT.22)GO TO 30

```

```

CALL SYMBOL(1.0,5.5,0.15,NAME(K),0.,40)

```

```

GO TO 40

```

30

```

CALL SYMBOL(.40,5.5,0.15,NAME(K),0.,50)

```

40

```

CONTINUE

```

C

C

C

C

C

C

C

```

MOVE PENCIL BACK TO ORIGIN

```

```

CALL PLOT(0.,0.,3)

```

C
C
C
C
C
C

POSITION THE PENCIL FOR THE NEXT PLOT

CALL PLOT(15.,1.,-3)

RETURN
ENDReproduced from
best available copy.

Appendix E

Computer Software (Trajectory Model)

This appendix contains the Fortran source code for the implementation of the various trajectories and the multiple hot spot projection model detailed in Chapter 2. The trajectories and hot spot locations were generated and saved on file for use in the computer simulation given in Appendix D. The program was written for use on the CDC Fortran IV compiler.

```

PROGRAM TRAJEC(INPUT,OUTPUT,TAPE3,TAPE6=OUTPUT,DEBUG=OUTPUT,TAPE8)
REAL UT(2,1),SVHS2A(200),SVHS2B(200),SVHS3A(200),SVHS3B(200)
REAL SVHT1A(100),SVHT1B(200)
REAL PULLUP
INTEGER IFRAME
INTEGER CON, OPLANE

```

```

C
C SET IIF=1 IF A PLOT OF THE INTENSITY CENTROID IS DESIRED
C IIF=2

```

```

C
C X0,Y0,AND Z0 ARE THE INITIAL INERTIAL COORDINATES
C X0=20000.
C Y0=10000.
C Z0=30000.

```

```

C
C SET CONT = 1 FOR THE 2G PULLUP AND CONTINUATION MANEUVER
C (TRAJECTORY 3)
C CONT=2

```

```

C
C SET OPLANE = 1 FOR THE OUT OF PLANE MANEUVER
C (TRAJECTORY 4)
C OPLANE=2

```

```

C
C DT=(1./30.)

```

```

C
C TSTART AND TFIN SET THE TIME FOR A ROLL TO START AND END
C TSTART=10.
C TFIN=11.

```

```

C
C IFRAME=155

```

```

C
C DISV IS THE DESIRED DISTANCE IN THE EBX DIRECTION FOR HOTSPOT
C ONE IN METER
C DISVP2 AND 3 ARE THE OFFSETS FOR HOTSPOTS 2 AND 3 IN THE
C EBY DIRECTION IN METER
C
C DISV=.1
C DISVP2=.1
C DISVP3=-.1
C TIME=0.

```

```

C
C PULLUP SETS THE TIME FOR THE PULLUP MANEUVER TO START
C (TRAJECTORY 2)
C PULLUP=10.0

```

```

C
C RATE SETS THE DEGREE OF THE PULLUP MANEUVER
C 2-G=.0196,5-G=.049,10-G=.098
C RATE=0.0196

```

```

C
C DO 90 NR=1,IFRAMES

```

```

C
C XDOT= -300.
C YDOT= -300.
C ZDOT= 0.
C XPOS= X0 + XDOT*TIME
C YPOS= Y0 + YDOT*TIME
C ZPOS= Z0 + ZDOT*TIME

```

Reproduced from
best available copy.

```

VTIME=TIME-(D1/2.0)
IF(TIME.LT.PULLUP)GO TO 310
IF(CONT.NE.1)GO TO 71
IF(NR.LT.106)GO TO 71
IF(NR.GE.106)XDOT=-999.510
IF(NR.GE.106)YDOT=29.063
IF(NR.EQ.106)TIMEA=1.
C THE X0 AND Y0 BELOW ARE VALID FOR A 2G PULLUP AT T=3.5 SEC
C WHERE THE TARGET CONTINUES AT A CONSTANT VELOCITY AFTER THAT POINT
C
X0=1500.2
Y0=52.205
XPOS=X0+XDOT*TIMEA
YPOS=Y0+YDOT*TIMEA
TIMEA=TIMEA+D1
GO TO 72
71 CONTINUE
IF(ORPLANE.NE.1)GO TO 73
XDOT=-1000.*COS(TRATE*(VTIME-2.))
YDOT=1000.*COS(TRATE*(VTIME-2.))*SIN(TRATE*(VTIME-2.))
ZDOT=-1000.*(SIN(TRATE*(VTIME-2.))*2)
XPOS=X0-2000.-((1000/TRATE)*SIN(TRATE*(TIME-2.)))
YPOS=Y0+((1000/(2*TRATE))*(SIN(TRATE*(TIME-2.))*2))
ZPOS=Z0-1000.*(((TIME-2.)/2.)-(1./(4.*TRATE))*SIN(2*TRATE*(TIME-2.)))
GO TO 310
73 CONTINUE
YDOT=-1000.*COS(TRATE*(VTIME-2.))
YDOT=1000.*SIN(TRATE*(VTIME-2.))
XPOS=X0-2000.-((1000/TRATE)*SIN(TRATE*(TIME-2.)))
YPOS=Y0+((1000/TRATE)*(1.-COS(TRATE*(TIME-2.))))
72 CONTINUE
ZPOS=Z0
310 CONTINUE
RHOR= XPOS**2 + ZPOS**2
RANGE= RHOR + YPOS**2
UT(1,1)= (-ZPOS*XDOT+XPOS*ZDOT)/(RHOR*.00002)
RHOR= SQRT(RHOR)
UT(2,1)=(RHOR*YDOT-YPOS*((XPOS*XDOT+ZPOS*ZDOT)/RHOR))/(RANGE*.00002)
RANGE= SQRT(RANGE)
VMAG=SQRT(XDOT**2+YDOT**2+ZDOT**2)
VMAX=(VMAG)/(RANGE*.00002)
CALL HSPOT(HS1A,HS1B,HS2A,HS2B,HS3A,HS3B,TSTART,TFIN,TIME,COSW
+ ,XPOS,YPOS,ZPOS,XDOT,YDOT,ZDOT,DISV,DISVP2,DISVP3,
+ NR,SVHS2A,SVHS2B,SVHS3A,SVHS3B,VMAG,RHOR,RANGE,SVHS1A,SVHS1B)
WRITE(NR,*)NR,UT(1,1),UT(2,1),VMAX,COSW,RANGE,TIME
80 FORMAT(I4,6E14.5)
WRITE(NR,*)HS1A,HS1B,HS2A,HS2B,HS3A,HS3B
81 FORMAT(6E14.5)
WRITE(NR,*)XPOS,YPOS,ZPOS,XDOT,YDOT,ZDOT,TRATE
83 FORMAT(7E14.5)
WRITE(NR,*)XPOS,YPOS,ZPOS,XDOT,YDOT,ZDOT,TRATE
82 FORMAT(1X,7E14.5)
TIME=TIME+D1
80 CONTINUE
C

```

IF(III.EQ.1)CALL PLTD(SVHS2A,SVHS2B,SVHS3A,SVHS3B,NFPAYES,SVHS1A,S
+VHS1B)

C
C

END

SUBROUTINE PLTD(SVHS2A,SVHS2B,SVHS3A,SVHS3B,NFRAME,SVHS1A,SVHS1B)

C

REAL SVHS2A(200),SVHS2B(200)
 REAL SVHS3A(200),SVHS3B(200)
 REAL SVHS1A(200),SVHS1B(200),HOLD1(600),HOLD(600)
 INTEGER CT,N3

C

N=NFRAME
 N3=NFRAME*3
 CT=1

C

DO 10 I=1,NFRAME
 HOLD(CT)=SVHS1A(I)
 HOLD(CT+1)=SVHS2A(I)
 HOLD(CT+2)=SVHS3A(I)
 HOLD1(CT)=SVHS1B(I)
 HOLD1(CT+1)=SVHS2B(I)
 HOLD1(CT+2)=SVHS3B(I)

10

CT=CT+3
 CONTINUE
 CALL PLOT(S(0.,0.,9)
 CALL PLOT(1.,1.,-3)
 CALL SCALE(HOLD,4.,3,1)
 CALL SCALE(HOLD1,4.,3,1)
 SVHS2A(N+1)=HOLD(N3+1)
 SVHS2A(N+2)=HOLD(N3+2)
 SVHS2B(N+1)=HOLD1(N3+1)
 SVHS2B(N+2)=HOLD1(N3+2)
 CALL AXIS(0.,0.,27H PROJECTION ON THE ALPHA AXIS,
 +27,4.,0.,SVHS2A(N+1),SVHS2A(N+2))
 CALL AXIS(0.,0.,27H PROJECTION ON THE BETA AXIS,
 +27,4.,0.,SVHS2B(N+1),SVHS2B(N+2))
 SVHS3A(N+1)=SVHS2A(N+1)
 SVHS3B(N+1)=SVHS2B(N+1)
 SVHS3A(N+2)=SVHS2A(N+2)
 SVHS3B(N+2)=SVHS2B(N+2)
 SVHS1A(N+1)=HOLD(N3+1)
 SVHS1A(N+2)=HOLD(N3+2)
 SVHS1B(N+1)=HOLD1(N3+1)
 SVHS1B(N+2)=HOLD1(N3+2)

C

CALL LINE(SVHS2A,SVHS2B,N,1,-1,3)
 CALL PLOT(0.,0.,3)
 CALL LINE(SVHS3A,SVHS3B,N,1,-1,2)
 CALL PLOT(0.,0.,3)
 CALL LINE(SVHS1A,SVHS1B,N,1,-1,4)
 CALL PLOT(-1.,-1.,-1)
 CALL PLOT(6.,0.,2)
 CALL PLOT(6.,6.,2)
 CALL PLOT(0.,6.,2)
 CALL PLOT(0.,0.,2)
 CALL SYMBCL(1.70,5.3,0.15,17H INTENSITY PATTERN,0.,17
 +)

C

CALL PLOT

C

RETURN

```

SUBROUTINE H-POC(H1A,HS1B,HS2A,HS2B,HS3A,HS3B,TSTART,TFIN,TIME
*,COSW,XPOS,YPOS,ZPOS,XDOT,YDOT,ZDOT,DISV,DISVP2,DISVP3,
*,SVHS2A,SVHS2B,SVHS3A,SVHS3B,VMAG,PHOR,RANGE,SVHS1A,SVHS1B)
REAL SVHS2A(200),SVHS2B(200),SVHS3A(200),SVHS3B(200)
REAL SVHS1A(200),SVHS1B(200)

```

C

C

C

C

C

```

*ROLL SETS THE RATE OF THE CONSTANT ROLL MANEUVER
FOR A 360 ROLL IN 4 SECS ROLL=1.2566
ROLL=.5

```

```

ALPHA=ATAN(ZPOS/XPOS)
BETA=ATAN(YPOS/PHOR)
VXZ2=(XDOT**2+ZDOT**2)
SQVXZ=SQRT(VXZ2)
CINB=SIN(BETA)
COSA=COS(ALPHA)
SINA=SIN(ALPHA)
COSB=COS(BETA)

```

C

C

```

HS1B=((DISV/RANGE)*((XDOT*SINB*(-COSA))+(YDOT*COSB)+(ZDOT*CINB*
* (-SINA))))/(VMAG*.00002)
HS1A=((DISV/RANGE)*((XDOT*(-SINA))+(ZDOT*COSA)))/(((VMAG)*.00002)
IF (TIME.GT.TFIN)GO TO 63
IF (TIME.LT.TSTART)GO TO 60
COSW=COS(ROLL*(TIME-TSTART))
SINW=SIN(ROLL*(TIME-TSTART))
63 DPPV=SQRT(XDOT**4+(YDOT**2)+(XDOT**2)+(2*(
* XDOT**2)*(ZDOT**2))+(ZDOT**2)*(YDOT**2)+(ZDOT**4))
GO TO 61
60 SINW=0.
COSW=1.
DPPV=1.

```

63

60

C

61

```

SAVEA=((COSW*(-ZDOT))/SQVXZ)+((SINW*YDOT*XDOT)/DPPV)
SAVEB=((SINW*YDOT*ZDOT)/DPPV)+((COSW*XDOT)/SQVXZ)
DELA=((SAVEA)*(-SINA))+(SAVEB*COSA)
DELB=(SAVEA*SINB*(-COSA))+((-VXZ2*SINW*COSB)/DPPV)
*+(SAVEB*SINB*(-SINA))

```

C

```

HS2A=((DISVP2/RANGE)*DELA)/.00002)
HS2B=((DISVP2/RANGE)*DELB)/.00002)
HS3A=((DISVP3/RANGE)*DELA)/.00002)
HS3B=((DISVP3/RANGE)*DELB)/.00002)

```

C

C

```

SVHS1A(NP)=H1A
SVHS1B(NP)=H1B
SVHS2A(NP)=H2A
SVHS2B(NP)=H2B
SVHS3A(NP)=H3A
SVHS3B(NP)=H3B

```

C

C

```

RETURN
END

```

Reproduced from
best available copy.

Vita

Paul P. Millner was born on January 31, 1952 in Danville, Virginia. He graduated from West Mecklenburg High School in Charlotte, North Carolina in June 1970 and entered the United States Military Academy at West Point in July of the same year. In June 1974, he graduated from West Point with a Bachelor of Science degree and was commissioned as an Army Air Defense officer. Military assignments include: Vulcan platoon leader and executive officer, 2nd Armored Division, Fort Hood, Texas; Redeye platoon leader, 2nd Infantry Division, Korea; Improved HAWK battery commander, 11th Air Defense Artillery Brigade, and instructor, Department of Tactics, United States Army Air Defense Artillery School, Fort Bliss, Texas. Captain Millner is a graduate of the Air Defense Artillery Officers Basic and Advanced Course. In June 1981, Captain Millner was assigned to the Air Force Institute of Technology to pursue a Masters Degree in Electrical Engineering.

Permanent address: 4226 Westridge Drive
Charlotte, N.C. 28208

60
62

00 65 J=1,
IF(J-K) 62,65,62
KJ=J-I+K

344

END

FILMED

3-83

DTIC